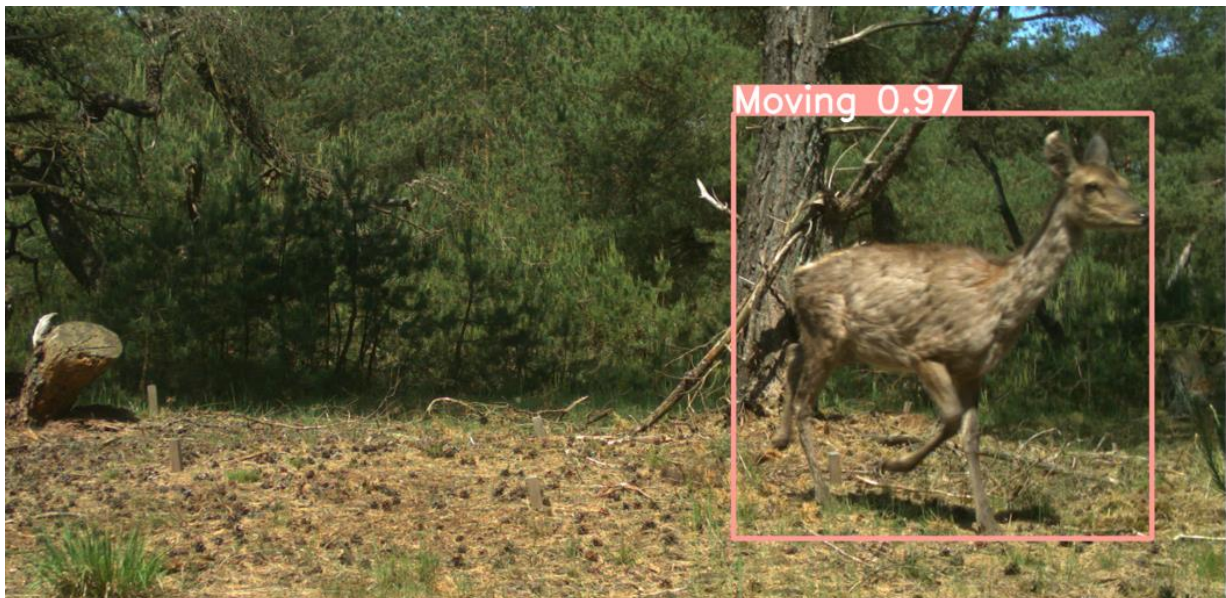


Recognition of wildlife behaviour in camera-trap photographs using machine learning

Jorrit van Gils

26/04/2022



Recognition of wildlife behaviour in camera-trap photographs using machine learning

A comparison of YOLOv5 and the pose estimation approach

Name course : Thesis Wildlife Ecology and Conservation

Number : WEC-80436

Date : 21/02/2022

Student : JN (Jorrit) van Gils

Registration number : 1006511

Study programme : MSc Forest and Nature Conservation

Supervisors : Dr. ir P.A. (Patrick) Jansen

Dr. ir H.J. (Henjo) de Knecht

External advisors : H. (Helena) Russello (Farm Technology)

Dr. G.W. (Gert) Kootstra (Farm Technology)

Drs. R. Hollands (Verrassend Hollands)

Group : Wildlife Ecology and Conservation Group

6708 PB Wageningen

T: +31 (317) 48 58 28

E: patricia.meijer@wur.nl

Disclaimer

This report is written by a master student Forest and Nature Conservation guided by the Wildlife Ecology and Conservation chair group. Using the information in this thesis is at own risk. It is highly recommended to check the information before it is applied. The University of Wageningen will not be responsible by law, for the consequences of this report. Reproduce or publish information from this report is only allowed with explicit approval of:

Wageningen University and Research

Wildlife Ecology and Conservation Group

6708 PB Wageningen

T: +31 (317) 48 58 28

E: patricia.meijer@wur.nl

Preface

The motivation for this MSc thesis 'Recognition of wildlife behaviour in camera-trap photographs using machine learning' is derived from a long-held passion on animal conservation. I would like to acknowledge and thank my supervisors dr. ir. Patrick Jansen and dr. ir. Henjo de Knecht for their expert guidance and support.

The research was challenging because the subject, largely about deep learning was not included as standard in the Forest and Nature Conservation curriculum. Fortunately, my external supervisors H. Russello, Dr. G.W. Kootstra and Drs. R. Hollands were always available and willing to answer my deep learning related queries.

Finally, I would like to express gratitude to, J.E. Doornweerd, D. M. Gonzalez, B. Jackson, R. Bloo, N. Heida, L. Dijkhuis, B. Sluiter, S. Beery, A. Maxwell and dr. J.H.M. Feyen for their wonderful cooperation as well.

List of abbreviations

<i>AI</i>	Artificial Intelligence
<i>AR</i>	Action recognition
<i>CNN</i>	Convolutional neural network
<i>CV</i>	Computer vision
<i>DL</i>	Deep learning
<i>DLC</i>	DeepLabCut
<i>FIG</i>	Feature importance graph
<i>GPU</i>	Graphics processing unit
<i>MAE</i>	Mean average error
<i>ML</i>	Machine learning
<i>MLP</i>	Multi layer perceptron
<i>NN</i>	Neural networks
<i>NPHV</i>	National Park the Hoge Veluwe
<i>OD</i>	Object detection
<i>PE</i>	Pose estimation
<i>PEA</i>	Pose estimation approach
<i>RF</i>	Random forest
<i>YOLO</i>	You only look once

Abstract

With more than a million cameras to monitor wildlife actions, human analysis of camera-trap data has become time-consuming and unsustainable. Deep learning (DL) algorithms that can automatically extract features from photographs, provides a powerful solution but are not widely used yet for automated wildlife action recognition (AR).

Using DL for wildlife presents the added challenge of a highly varying background, causing models that classifies objects based on full images to often predict well on training data but poorly on new test data. As new images still contain the same species, but in a different background, models that classify based on the entire photo seem to experience distraction from the background. DL object detection (OD) methods have come up with an innovative way to reduce or overcome background distortion by classifying based on the localization of the animal. Two OD methods have grown in popularity especially in recent years; YOLOv5 and Pose estimation. How these models perform on determining wildlife actions is not much explored.

In a case study on Red deer (*Cervus elaphus*) in National Park the Hoge Veluwe in The Netherlands YOLOv5 and Pose Estimation were applied to examine their predictive capacity. A relatively small dataset of 506 single Red deer images labelled with actions *foraging*, *moving* or *other* was created. YOLOv5 classified the action of an animal based on a bounding box framed around the animal. In the pose estimation approach (PEA), the action was classified by a RF algorithm that used key-point features as input extracted from pose estimation technique DeepLabCut (DLC). Both methods were evaluated for accuracy, prediction, recall and human-effort. During comparison methodological differences were considered.

It was hypothesized that PEA compared to YOLOv5 would reach higher reliability than pose estimation as less training data was needed and it had fine-grained control by only looking at the relevant properties of an object. Contrary to expectations, YOLOv5 outperformed PEA by achieving a higher accuracy (0.55) compared to PEA (0.53), being six times less time consuming, requiring less computational power and by user-friendly inference in which you apply the model to new data. Predictive capacity of YOLOv5 is mainly expected to improve by increasing the dataset class sizes. Although results point in favour of the YOLOv5 model, there is a suspicion that PEA does have a lot of potential, but in hindsight the sub-optimal set-up of annotated key-points obscured by the animal and using the full image as input to DLC may have led to a lower predictive capacity. It should be further investigated how much better PEA performs compared to YOLOv5 when improving the labelling strategy and when the bounding box is used as input for DLC. While these individual AR models are already a great step forward for conservation, for a more extensive applicability it is advised to refine the models with a greater library of pictures containing multiple Red deer individuals and by classifying behaviour based on animal's action changes of successive images.

Table of contents

1. Introduction	8
1.1 Introduction	8
1.2 Research questions	9
2. Methods	10
2.1 Study system	10
2.2 Taxonomy of actions	10
2.3 Dataset creation	12
2.4 Object detection	14
2.4.1 YOLOv5	14
2.4.2.1 Box 1. Setting up YOLOv5 via BOX21	14
2.4.2.2 Box 2. Origin and how YOLOv5 works	16
2.4.2 The pose estimation approach	17
2.4.2.1 DeepLabCut	17
2.4.2.2 Random Forest	18
2.4.2.3 Box 3. The history of pose estimation	20
2.4.3 Background	20
2.4.3.2 Box 4. The history of deep learning	20
2.4.3.2 Box 5. Automated action recognition of wildlife	22
3. Results	24
4. Discussion	27
5. Reference list	30
Appendix 1: Map National Park Hoge Veluwe	34
Appendix 2: Action annotation protocol	35
Appendix 3: Scripts and models	36
Appendix 4: BOX21 example input	37
Appendix 5: Train and test error graphs	38
Appendix 6: DLC annotation protocol	39
Appendix 7: PEA key-point features and distribution	42
Appendix 8: Results	43

1. Introduction

1.1 Introduction

Studying the behaviour of wild animals is a topic of great importance to conservationists (Braude, Margulis, & Broder, 2017). Knowing animal behaviour helps to understand their requirements, its habitat, needs, preferences and dislikes (Mench, 1998). Providing ecologists with accurate and extensive knowledge about animal behaviour can help to inform ecology and conservation management (Norouzzadeh et al., 2018). Action recognition (AR) refers to identifying what an animal is doing in an image (W. Li, Swetha, & Shah, 2020) and is the foundation for understanding the animals behaviour (Lehner, 1992).

Over the past decades techniques to recognize wildlife actions have changed rapidly from using ethograms (Altmann, 1974) to the use of animal electronic sensors, that send signals that can be picked up by a receiver (Lagardère, Anras, & Claireaux, 1999). A constrain to both of these methods is that the observer has to be in the vicinity of the animal before or during the observation (Schneider, Taylor, Linqvist, & Kremer, 2019). Electronic sensors were considered laborious, expensive (Schneider et al., 2019), harmful (Mellor, Beausoleil, & Stafford, 2004) and displacement data is of no use in monitoring animals actions. Most importantly, tagging was unable to scale to large populations (X. Li, Cai, Zhang, Ju, & He, 2019).

Camera-traps are an alternative way to extract actions from the data. Camera-traps provide a window into the animals world, with lower costs and reduced workload for researchers (Braude et al., 2017; Norouzzadeh et al., 2018; Schneider et al., 2019) and therefore are among the most used sensors by ecologists (Tuia et al., 2022). With the current rate of acquiring images being extremely large and overwhelming automation of AR is essential to allow for large scale analysis (Pereira et al., 2019; Schneider et al., 2019).

The standardized and automated alternative to manual image analysis is computer vision (CV) (Schneider et al., 2019), in which computers are programmed to interpret the visual world (A. Zhang, Lipton, Li, & Smola, 2020). Early CV models were restricted as they could only recognize parts of an object separately and manual calculations of relative distances between these parts were required to make a binary choice whether something was an object or not (Fergus, Perona, & Zisserman, 2003).

In recent years CV has evolved rapidly due to deep learning (DL), which is a subdivision of machine learning (ML) which in turn is part of artificial intelligence (AI). ML deals with providing computers the ability to learn, without being explicitly programmed (Hastie, Tibshirani, Friedman, & Friedman, 2009). DL evolved from ML by having multi-layered neural networks (NN), that attempted to simulate the action of the human brain. These DL-models turned out to be extremely powerful because a neural network is designed in such a way that it can approximate any function to fit the data (Kratsios, 2021).

As DL algorithms include automated feature extraction, manual intervention was not needed anymore to calculate features like the distance between the muzzle and the toe, which can improve objectivity and remove human bias in which researchers assume what features are important (Schneider et al., 2019). Automated feature extraction can be interpreted by the network recognizing the presence of small- or large objects, the position of objects in the image or the colour contrast. Although automatic feature extraction came at the expense of human reasoning and interpretability, with DL models being often referred to as a black box (Waldrop, 2019), DL algorithms often outcompeted traditional ML methods especially when using large datasets (Freytag et al., 2016; Schneider et al., 2019).

Despite its potential, DL is particularly challenging when applied to wildlife-camera-trap data because of the uncontrolled varying backgrounds (Ravoor & Sudarshan, 2020). DL algorithms risk to learn

patterns including the background, rather than a 'visual concept' of animals action that can be applied to new data (Beery, Van Horn, & Perona, 2018). Models that perform classification based on full images often prove to not be applicable in practice as, whilst in test environments they are able to make predictions based on images encountered, the varying reality of different background and light conditions has proven to deliver less accurate predictions (Tuia et al., 2022).

Therefore, DL models that lend itself to more accurate evaluation are object detection (OD) models (Michelucci, 2019). Instead of classification based on the full images, these models focus only on the relevant properties of the animal (Kubat & Kubat, 2017; Michelucci, 2019). As OD models experience less background disturbance, they could provide a powerful solution against overfitting of the background, and therefore creating a model also suitable for predictions on new data (Beery et al., 2018).

YOLOv5 (Bochkovskiy, Wang, & Liao, 2020) used in recognition is currently a popular method as it can very quickly and accurately detect and classify objects based on a bounding box (Bochkovskiy et al., 2020; Michelucci, 2019). Alternatively pose estimation method DeepLabCut (DLC) extracts key-points from animal poses and is well liked as it only outputs the relevant properties of the animal (Mathis et al., 2018). The aim of this study was to find out which of the two algorithm works best for wildlife AR.

While YOLOv5 directly outputs an action, DLC contained key-points as output, requiring an extra step to recognize the action. Therefore, after DLC, actions were classified by applying a random forest (RF) algorithm that used key-points as input, creating a two-step method called the pose estimation approach (PEA). It is hypothesized that PEA would lead to a better predictive performance because of its ability to generate synthetic key-points and fine-grained control by only looking at the relevant animals key-points (Pereira, Tabris, et al., 2020). PEA is especially interesting because it is stated that DLC can already make accurate predictions with a limited number of training data (Nath et al., 2019; Pereira, Shaevitz, & Murthy, 2020; Pereira, Tabris, et al., 2020) while for YOLOv5 a greater volume of images are needed as multiple permutations of each pose needed to be collected. In addition to predictive performance, also feasibility of implementing the methods is considered. As pose estimation technique DLC requires more research, has a longer estimated training time, and requires labelling of animal key-points for every training image, PEA is likely to be more time-consuming than YOLOv5.

As a case study, 506 single Red deer (*Cervus elaphus*) images, are used, collected by camera-traps scattered across National Park Hoge Veluwe (NPHV), a 50-km² game reserve in the Netherlands. The goal of this study is to apply and compare two leading OD methods (YOLOv5 and the pose estimation approach (PEA)) for Red deer AR and evaluate how well they adapt to new data, while taking into consideration the level of human effort in creating the methods.

1.2 Research questions

RQ 1: How do the methods YOLOv5 and the pose estimation approach (PEA) perform?

RQ 2: How feasible are the two methods in terms of human effort to build and implement?

2. Methods

2.1 Study system

National Park Hoge Veluwe (NPHV) is a fenced nature reserve with a total area of 50 km² located in the province Gelderland of the Netherlands (Appendix 1: Map National Park Hoge Veluwe). NPHV has a temperate climate with relatively mild winters and summers (Grieser, Gommel, Cofield, & Bernardi, 2006). NPHV covers 5% of the Veluwe which is the biggest contiguous nature reserve of the Netherlands and is visited by over half a million people each year. Six diverse landscapes are present, resulting from historical land use: drift sand, meadow, dry and wet heathland, pine and oak forest (National Park Hoge Veluwe, 2021).

NPHV is inhabited by many large mammals, including Mouflon (*Ovis gmelina*), Roe deer (*Capreolus capreolus*), Wild boar (*Sus scrofa*) and Fallow deer (*Dama dama*) and approximately 200 Red deer (*Cervus elaphus*) individuals. Red deer belongs to the clade of ungulates characterized by walking on hooves (toes) with their heel not connected to the ground (Janis, 1998). Red deer is an herbivorous species and their diet consist of grass, heather, leaves, buds and shoots (National Park Hoge Veluwe, 2021). It is one of the largest deer species in the world and can be found in most of Europe. The animals length is between 160cm and 250cm and their weight varies from 120kg to 240kg with males usually larger than females (National Park Hoge Veluwe, 2021). Red deer have several prominent and/or contrasting body features like slender legs, a thin face, pointy ears and a black muzzle. The orangish to brown coloured skin makes them less noticeable in their environment (Ratray, 2009).

In NPHV, 70 permanent camera-traps continuously record animal activity, with at least eight camera stations per habitat type (National Park Hoge Veluwe, 2021). The animals are recorded with wildlife camera-traps (HC500, RECONYX, Holmen USA) that are placed on poles 70 cm above the grounds surface. The camera-trap data includes variation in lighting conditions by recording 3.1 MP colour images (RGB) during daylight and monochrome black and white images in night-time. Camera-traps that detect an animal with their passive infrared heat sensor shoots a sequence of 10 images with an interval of 0.9 seconds. When the camera is triggered again shortly after, another 10 images are recorded and added to the sequence.












The NPHV camera-trap dataset with content from 2013 and onwards contains half a million sequences and more than 6 million images including metadata like the URL, location, time, sequence and unique multimedia identifier (Cameratrap DP Development Team, 2021). Volunteers, students and researchers worked on labelling images, for example with the type of species, the amount of animals, sex and age (Casaer, Milotic, Liefing, Desmet, & Jansen, 2019).

2.2 Taxonomy of actions

11 Red deer actions have been detected after image analysis and literature review. Red deer are intermediate foragers that *graze*¹ (on grasses), *browse*² (on hardwood vegetation) and *scan*³ (attempting to forage) (Gebert & Verheyden-Tixier, 2001). Red deer gait can be distinguished by *walking*⁴ generally characterized by having mostly 3 legs in contact with the ground and *running*⁵ with maximum 2 legs in contact with the ground (Wada, 2022). The sixth and seventh action are *sitting*⁶ on the ground and *standing*⁷, in which the ears are low. More often however, *sitting* or *standing* is accompanied by *vigilance*⁸, in which the animal is on her guard with upward pointing ears (Ratray, 2009). The ninth action is *grooming*⁹ in which the animal cleans its fur to remove insects and parasites (Graystock & Hughes, 2011). Typical courting behaviour of males is *roaring*¹⁰ (National Park Hoge Veluwe, 2021). Animal responses to the sound of camera-traps recording images causes a non-natural eleventh action which is *watching into the camera*¹¹.

The above-described actions resulted in 11 Red deer actions (Table 1). To ensure concise action labelling for all images during dataset creation, an action annotation protocol was drawn up that included all 11 actions (Gils, 2021) of which the action labelling criteria are included in this report (Appendix 2: Action annotation protocol).

Table 1 The 11 different forms of Red deer action, from <https://www.agouti.eu/>.

1.Grazing 	2.Browsing 	3.Scanning 	4.Walking 
5.Running 	6.Sitting 	7.Standing 	8.Vigilance 
9.Grooming 	10.Roaring 	11.Camera watching 	

Labelling these actions unavoidably contained several limitations. First, although it was possible to expand action *running* by trot, pace and gallop, it was decided not to do this because they could be open to interpretation and a common dataset was to be immediately understood by all annotators. Second, as defecating or urinating were not observed in the dataset, these behaviours were excluded from the action protocol. In addition, social actions were excluded as this research focussed on recognizing individual actions. Third, limitation to this human labelling protocol was that inconsistencies between analyses might still occur as annotators do not always interpret these criteria the same way despite the extensive description of actions (Schneider et al., 2019). Therefore, special attention has been paid to very precisely specifying the criteria of classes which are mostly confused like *walking* and *running*, *scanning* and *moving*, *scanning* and *grazing/browsing* and *vigilance* and *standing*. Finally, it is recognised that limitations to the action classification can also be a result of basing conclusions on single images instead of consecutive images. Whilst this made it more manageable to create the models, it reduced reliability of labelling considering actions are a dynamic and temporal activity (Pereira et al., 2019). For example, the differences in *walking* and *running* characterized by contact points with the ground was best observed from consecutive frames but had to be distinguished from a single image.

2.3 Dataset creation

The data was retrieved from Agouti (agouti.eu), a platform for managing and processing camera-trap data and filtered in R (R Core Team, 2020). As all images required action-labelling, key-point labelling and processing through the 2 approaches, the number of images that could be dealt with was limited to approximately 500 images due to time constraints. After registration and having access to the project ‘the Hoge Veluwe wildlife monitoring project’, unzipping the data ‘19 October 2021’ resulted in three files: *observations.csv*, *multimedia.csv* and *deployments.csv* that were loaded into R. Filters were applied for species Red deer (1) and a single animal (2), reducing the dataset from 449.575 to 6.895 sequences (Table 2, part 1).

Table 2-part 1 Initial filtering for single red deer.

	Filter	Sequences
	<i>Initial Aguti dataset 2013- 2021</i>	449.575
1	Species Red deer	12.562
2	Single Red deer	6.895

To measure how the model worked for new data, independency between the train and test dataset was essential (Beery et al., 2018). Therefore, each set was assigned to a different set of camera-trap locations (3) and segregated between pre and post 2019 (4) (Table 2, part 2). In addition, to prevent repetitive data within each set, the maximum number of sequences per year, per camera-trap location was set to 10 (5). Afterwards the available sequences were split into a 700 train (80%) and 175 test (20%) images, based on consulted literature (Bunkley, 2009).

Table 2-part 2 Creating independent train and test dataset

	Filter	Train	test
3	Camera-trap location	Set A	Set B
4	Year	< 2019	≥ 2019
5	Maximum	Per year and per camera location max 10 sequences	Per year and per camera location max 10 sequences
		700 sequences	175 sequences

Only images with a single animal fully covered in the frame were included (6) (Table 2, part 3), with a maximum of one image per sequence (7) aimed to increase the number of rare actions (*browsing*, *roaring*, *sitting* and *camera watching*). Red deer images obscured by vegetation were included to let the network learn predicting obscured body parts. Of note is that, pre-testing the YOLOV5 bounding box detector revealed some images contained multiple animals that were hard to detect with the human eye. Therefore, an extra 19 images were removed (8) ending up with a start dataset of 506 images, which the numbers matched the pre-intended goal of processing up to 500 images. Of this start dataset 394 images belonged to the training and 112 to the test set.

Table 2-part 3 From image sequences to usable individual images

	Selection	Sequences	images
	<i>Usable sequences</i>	875	
6	Manual criteria selection	556	
7	Manual sequence to image selection		556
8	Delete multiple animals bounding box		506 -> 394 train, 112 test

No filtering was applied for variation in animals age, viewpoint and shape, habitat type and lightning conditions like day or night, ensuring variation on these characteristics in the dataset (Beery et al., 2018). Given it was stated that, for pose estimation technique DLC, 100 labelled images could deliver

a lower than 5 pixel error (Nath et al., 2019), the training set of 394 images seemed sufficient to obtain a good key-point prediction. However, when these 394 images for action classification were divided over the 11 actions, with on average 35 images per action, a problem arose because this number was way to less to capture as many animal viewpoints from different angles, as it was recommended to have at least 150, but preferably 500 or more images per class for a reasonable accuracy (Shahinfar, Meek, & Falzon, 2020). This meant to train a reliable model out of the 394 training images, only a maximum of three action classes could be created.

The issue of most classes being too small was addressed by merging either a small class with a bigger class or multiple small classes into a single large class (Figure 1). For example, as *scanning* and *browsing* contained only respectively 43 and 7 images, it was determined these could be merged with the 134 images of *grazing* to form the large enough and logical parent class of *foraging* (184). Likewise, in recognition that single images made it difficult to distinguish *running* and *walking* classes, these were merged to form the large enough and logical parent class *moving* (171). With the remaining six classes, each class with limited images, these were merged to create parent class *other* (151) that met the minimum class size requirement of 150 images (Shahinfar et al., 2020). That all classes contained roughly the same number of images was important to prevent models becoming biased towards the classes with more images (Norouzzadeh et al., 2018).

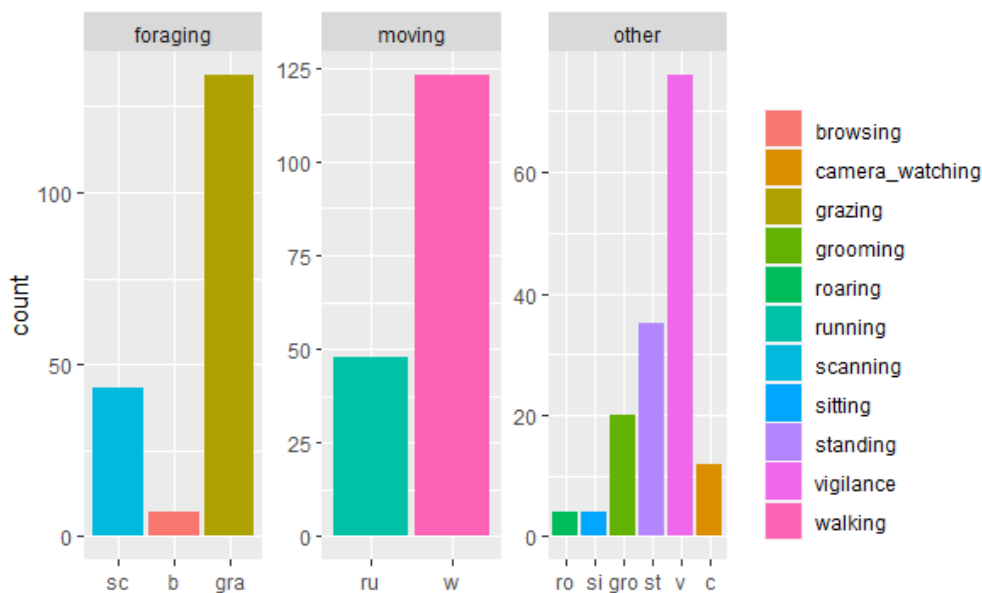


Figure 1 Distribution of the chosen Red deer actions (Wickham et al., 2019)

In R, all 506 images were labelled first with all 11 actions and then also with the 3 parent actions *foraging*, *moving* and *other*, according to the criteria in the action annotation protocol (Gils, 2021). The result of dataset creation was a csv file containing the filenames, action labels and the metadata (Table 3). The time effort in the dataset creation is shown in Table 4. From here programming shifted from R to Python used via the community edition of PyCharm (Van Rossum & Drake Jr, 1995), as Python better supported possibilities to apply ML and DL.

Table 3 Result dataset creation csv file with the attributes behaviour (parent actions) and behaviour_sub (all actions).

i	file_name	behaviour	behaviour_sub	in_validation_set	multimedia_id	path	deployment_id	sequence_id	location_name
1	00a6d9f6-0d07-4043-e110-ae5c308067e.jpg	m	w	FALSE	00a6d9f6-0d07-4043-e110-ae5c308067e	https://multimedia.agov.nl/assets/00a6d9f6-0d07-4043-e110-ae5c308067e	2cc8e513-ba9c-43c6-b050-0254f909e0d5	46xc55c-0090-41d0-b0d47-e37365b8651c	CB-H4-05
2	00b5636b-9c22-4249-bc85-d1b76934f93c.jpg	m	w	FALSE	00b5636b-9c22-4249-bc85-d1b76934f93c	https://multimedia.agov.nl/assets/00b5636b-9c22-4249-bc85-d1b76934f93c	7636a12d-8365-404a-b137-558a10d71b6683	c18266bbb-c77e-4e0e-b984c-c7ba4f618a1f	H3R0-02
3	00c62066-9c23-499f-8721-c0932c4e1f16.jpg	o	c	FALSE	00c62066-9c23-499f-8721-c0932c4e1f16	https://multimedia.agov.nl/assets/00c62066-9c23-499f-8721-c0932c4e1f16	719c4509-e00c-44aa-b206-0c01091f6b60	6f5492b-332b-410c-984c-83b077f761c2c	CB-H1-03
4	00f45ba7-3102-4e0c-b8c7-ab055000669a.jpg	o	st	FALSE	00f45ba7-3102-4e0c-b8c7-ab055000669a	https://multimedia.agov.nl/assets/00f45ba7-3102-4e0c-b8c7-ab055000669a	3b1473a3-3240-490d-a83e-b07901334c6c	49e49f9-2a1d-461e-9980-50d1c74b0d293	H6R0-03-V07
5	01154782-5f02-493f-80c5-8269c0085ce8.jpg	m	w	FALSE	01154782-5f02-493f-80c5-8269c0085ce8	https://multimedia.agov.nl/assets/01154782-5f02-493f-80c5-8269c0085ce8	4444c406e-7677-4890-b007-8a21930250b3	201180C10-0740-4795-b048-304ac68123f5	HER0-01

Table 4 Time effort dataset creation

Dataset creation	Estimated time in hours
Filter the Agouti dataset	1
Downloading Agouti images	7
Manual image selection	6
Annotating 506 images action	8
<i>Total</i>	<i>22</i>

2.4 Object detection

I applied and compared two object detection (OD) methods to the start dataset: the YOLOv5 and the pose estimation approach (PEA)(Figure 2).

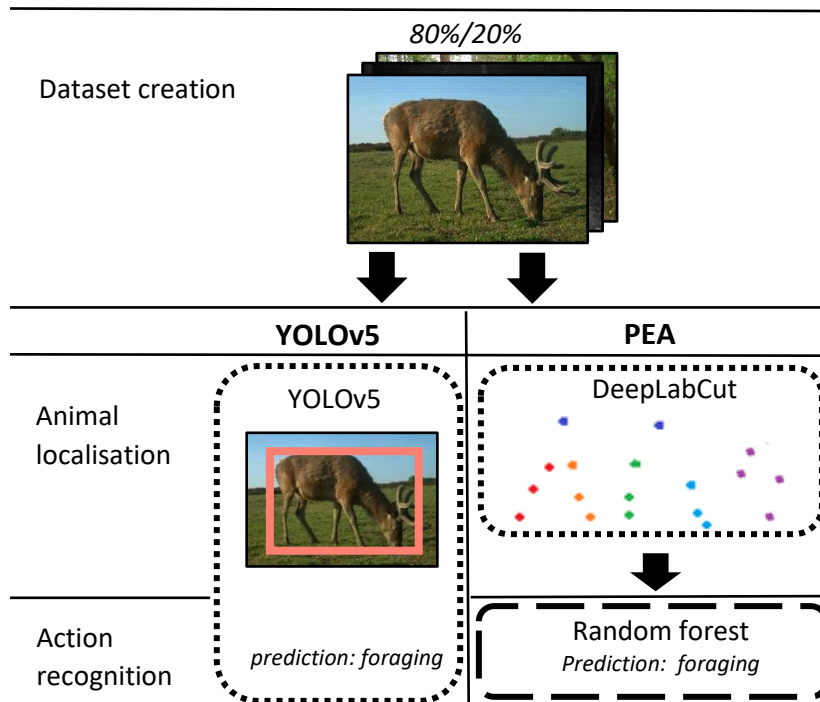


Figure 2 Overview of YOLOv5 and the pose estimation approach (PEA). Square outlines indicate a DL algorithm and the dashed outline a traditional ML algorithm

2.4.1 YOLOv5

YOLOv5 performs animal localisation and action recognition (AR) simultaneously (Michelucci, 2019). In this research YOLOv5 was applied with the GUI-driven workflow BOX21, in which users process steps by a point-and-click without the need for programming. On several occasions, it was necessary to switch to programming with code in Python (Appendix 3: Scripts and models), because this offered more possibilities to adjust.

2.4.2.1 Box 1. Setting up YOLOv5 via BOX21

BOX21 was accessed by setting up a tunnelling connection to the Wildlife Ecology and Conservation Graphics processing unit (GPU) PC at Wageningen University's Lumen building (RTX3090 24 GB, Nvidia, Santa Clara USA) via the Any Desk software which allowed the local PC to connect to the GPU PC, on which BOX21 was running. This GPU was needed because training a DL neural network is a resource-intensive task, and a GPU can perform multiple, simultaneous computations. After the connection was

established, BOX21 was accessed by typing *localhost:8008* in the web browser. As BOX21 required a different representation of the input data, the format of the start dataset csv had to be adjusted (Appendix 4: BOX21 example input). All uploaded images were manually checked to make sure every image contained one bounding box. In addition, these bounding boxes on BOX21 had to be linked to the already annotated action labels, which was done by specifying the labels again in the configuration file, selecting bounding boxes based on 'class like' and change the bounding box labels to the original labels.

The hyperparameter *number of epochs* was tuned which described when the entire dataset was passed through the network (Hastie et al., 2009). Setting the *number of epochs* too low risked not having enough time to learn. On the other hand, setting it too high will risked being too adapted to the training dataset increasing on the test error. Graphs that printed the errors for the train and test dataset showed that approximately 300 epochs was an optimum value (Appendix 5: Train and test error graphs). The amount of images processed per epoch is called the *batch size* and was set to 16 to find balance between a too low (not reaching optimum) and too high (overfitting) *batch size* value (Hastie et al., 2009).

As the start dataset contained limited class sizes, specifying hyperparameter *image augmentation*, in which new training images were created artificially, could play an important role to prevent overfitting (Michelucci, 2019). Image augmentation methods can prevent overfitting by increasing complexity of the model during training, which produces a higher training error, but can lower the error on the test dataset (Michelucci, 2019). For YOLOv5 *image augmentation* argument mixup was applied, that extends the training dataset by adding combinations of images (Figure 3) (H. Zhang, Cisse, Dauphin, & Lopez-Paz, 2017). Finally, the *confidence threshold* was set. Initially, the model does not output a specific class but instead assigns a confidence value between 0 and 1 for each class (Norouzzadeh et al., 2018). For example, setting the confidence threshold to 1 means the model only includes predictions when it is 100% certain removing all predictions with lower confidence values. A high confidence threshold can result in no bounding box being produced, although for each image a bounding box was labelled, creating a false negative. On the other hand, for a low confidence threshold, the model can predict multiple bounding boxes, which are called false positives. The confidence threshold of 0.5 was chosen to find a compromise between the number of false negatives and false positives. All hyperparameters were: *network: YOLOv5, epochs: 300, batch size: 16, image augmentation: mixup 0.5, confidence threshold: 0.5*

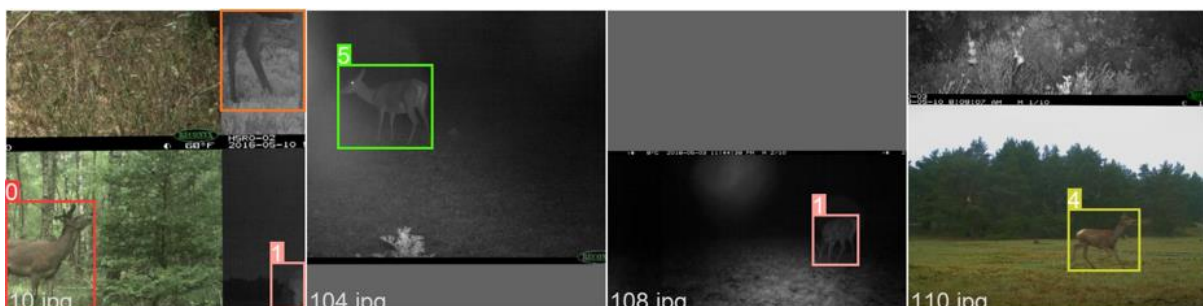


Figure 3 visual interpretation of the mixup argument in the YOLOv5 method

The first YOLOv5 model was trained with the 3 parent actions *foraging*, *moving* and *other*, whereafter a second model was trained that included all 11 actions, even though it was known that this second model would be less reliable. Predictive performance was measured using accuracy, precision and recall. Accuracy showed the percentage of correct predictions by the model, precision the percentage

of a predicted class that is confirmed that class and recall the percentage of all labels of a class that have been correctly classified. Precision values were also used to create the confusion matrix.

The capability of applying a model to new data is called inference and could be done either via BOX21 or programming in Python. For instance, inference via BOX21, the user can either upload new data similar as described earlier during training (Appendix 4: BOX21 example input) or leave out (or turn to False) the parameter *in validation set*. The active model can be run by navigating to *assets*, selecting the images and clicking *run model on selection*. Alternatively, download the best.pt model file from *models* for both models (Appendix 3: Scripts and models). Specify the URLs of the images you would like to predict and the path of your model file and obtain the prediction (Figure 4). Also, the time effort of YOLOv5 was included (Appendix 8: Results).



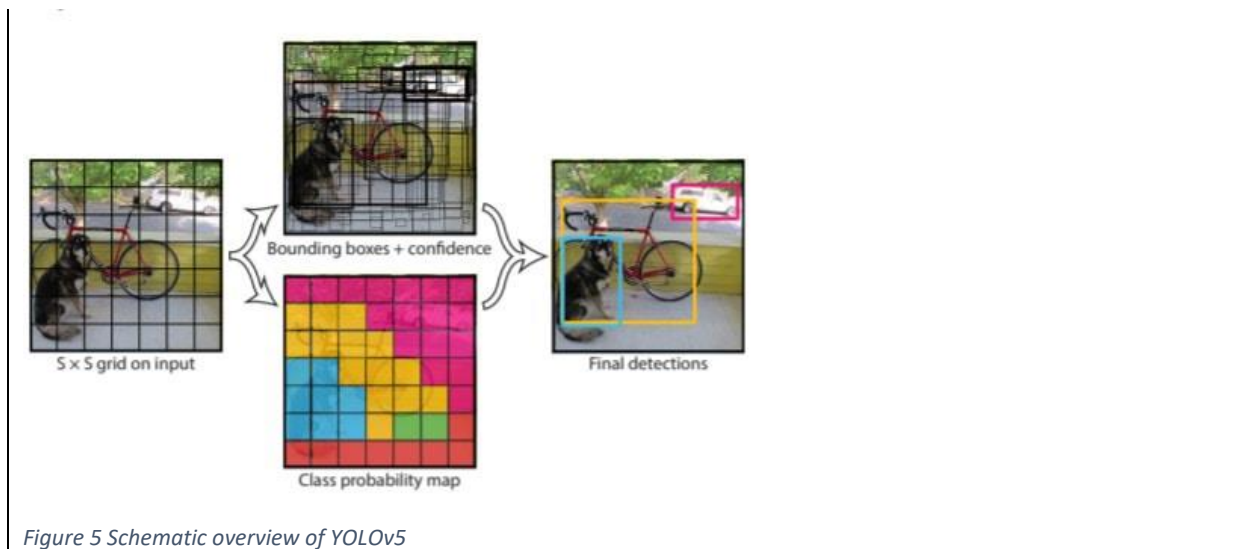
Figure 4 New data image from dataset BSc. student B. Sluiter can be classified via inference.

2.4.2.2 Box 2. Origin and how YOLOV5 works

A new family of object detection (OD) models, YOLO which stood for You Only Look Once, emerged in 2015 (Redmon, Divvala, Girshick, & Farhadi, 2016). Since 2020 the fifth version of the YOLO models, YOLOv5 are available (Bochkovskiy et al., 2020) and this network turns out to be one of the best OD models currently in circulation. Because YOLOv5 uses the full image as context for predicting bounding boxes (Redmon et al., 2016), YOLOv5 is much more efficient compared to previous methods like Fast R-CNN that used a sliding window approach in which all proposed object regions were required to go through the neural network (Girshick, 2015). YOLOv5 is also faster and more accurate than competing OD models like EfficientDet (Michelucci, 2019; Tan, Pang, & Le, 2020).

YOLOv5 is a convolutional neural network consisting of convolutional layers with at the end a fully connected layer. The YOLOv5 network was pretrained on the coco dataset consisting of 328.000 images (Lin et al., 2014). Applying this pre-trained network to a classification task with relatively few samples is called transfer-learning and can be useful as optimized neural network parameter values from a pre-trained neural network can be also explanatory for Red deer actions (Lin et al., 2014; Pereira et al., 2019).

YOLOv5 model first divides the image into a grid and predicts for every cell in the image a class confidence of the object, like 0.87 for a *foraging* Red deer. Then the network predicts several bounding boxes (Figure 5). The model is trained to predict the correct bounding box by evaluating the area of overlap between a predicted bounding box and a true labelled bounding box. The final detection combines the best fitting bounding box and adds the prediction of the class with the highest confidence value (Michelucci, 2019).



2.4.2 The pose estimation approach

In the pose estimation approach (PEA), first perform key-point localisation is performed using DeepLabCut (DLC), whereafter an AR random forest (RF) classifier is applied.

2.4.2.1 DeepLabCut

Many implementations for animal pose estimation (PE) have been developed, like DLC (Mathis et al., 2018), DeepPoseKit (Graving et al., 2019), LEAP (Pereira et al., 2019) and SLEAP (Pereira, Tabris, et al., 2020). For this research DLC was chosen because of its high performance and large community forums in which to be able to ask any questions (Nath et al., 2019).

DLC was applied with a GUI-driven workflow, however, on several occasions it was necessary to switch to programming with code in Python. 18 key-points assumed to maximize variation of each action were chosen on parts of the body that represented key-points while also being clearly visible from the outside. Visible key-points and key-points obscured by objects such as vegetation were labelled. Only key-points obscured by the animal itself, for example the knees on the non-visible side of the animal, were not labelled because the reasoning was that these key-points were often harder to predict (Figure 6) (Appendix 6: DLC annotation protocol). Labelling took approximately 2 minutes per image and thus, the total estimated time for labelling all images was 17 hours, which was performed in parts of half an hour, to ensure accurate annotation.

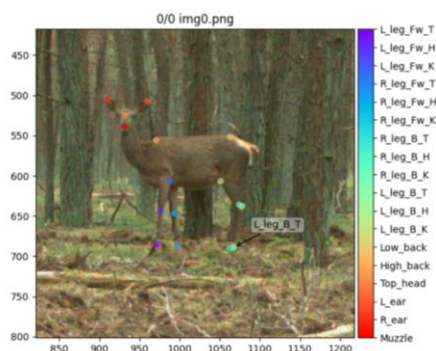


Figure 6 Annotation example from the labelling toolbox of DLC

Also, for PE technique DLC the hyperparameters were specified. The default *network* of DLC resnet50 was chosen which in PE typically consisted of an encoder and a decoder part. The encoder, extracted

features from the image and the decoder used this information to predict the key-point coordinates (Mathis, Schneider, Lauer, & Mathis, 2020). The *number of epochs* was set to 250.000 based on the train- test error graphs (Appendix 5: Train and test error graphs) and the *batch size* was set to 4. Similarly to mixup used in YOLOv5, here *image augmentation* method *imgaug* was applied which added training data by cropping, shifting, rotating and adding contrast (Nath et al., 2019). All hyperparameters were: *network: resnet_50, epochs: 250.000, batch size: 4, image augmentation: imgaug*

The models predicted key-points for both train and test dataset images. The margin of error was expressed in mean average error (MAE) which described the average pixel distance between the predicted key-point and its true label. With a training MAE of 23.31px and a test MAE of 217.45px, the model does largely overfit, despite applying mixup. Because DLC always predicts all-key-points, the white arrows in the key-point prediction plots (Figure 7) show that DLC had trouble predicting not labelled key-points (obscured by the animal itself) as the plots show these key-points became outliers predicted on completely different body parts. Not labelling these key-points, in retrospect, probably was not a sensible annotation choice for this study and might have had a considerable influence on the train and test error of the AR.

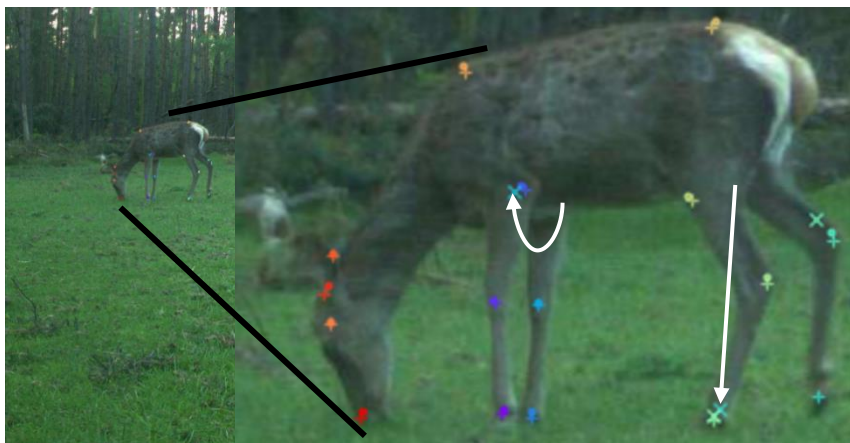


Figure 7 Example key-point prediction plot training test dataset. Labels ("+"), confident prediction ("•") and prediction("x"). White arrows shows the labelled key-point and the DLC prediction.

2.4.2.2 Random Forest

From the DLC key-point coordinates key-point features were calculated in such a way that they would explain the actions *foraging* and *moving* (Figure 8). Features 1 and 2 related to the legs angle and were created by using the math package (Van Rossum, 2020) while feature 3 and 4 related to the relative distances between the head and the leg (Figure 8). A description of the key-point features and how these key-point features are distributed is shown (Appendix 7: PEA key-point features and distribution).

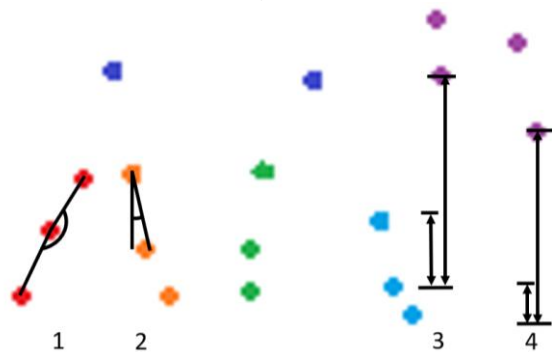


Figure 8 Visualisation of the calculated key-point features. Dots represent Red deer key-points.

As key-point features 3 and 4 contained large outliers, per key-point feature, values lower than the third percentile were changed to the key-point features 3 percentile value and values higher than the 97th percentile were changed to the key-point features 97th percentile creating a cut-off point off 3-97% for key-point feature 3 and 4. As many algorithms perform better when the range of the values between the features are equal, the features were normalised subtracting the values for each feature from its mean and scale the values to a standard deviation of 1 (Hastie et al., 2009).

Plotting a legs angle key-point feature (feature 1) and a head to leg distance key-point feature (feature 3) in relation to the left back leg for the classes *moving* and *foraging* revealed the head to leg distance features contained more variation between the two classes (Figure 9) and thus might be important in explaining variation in actions.

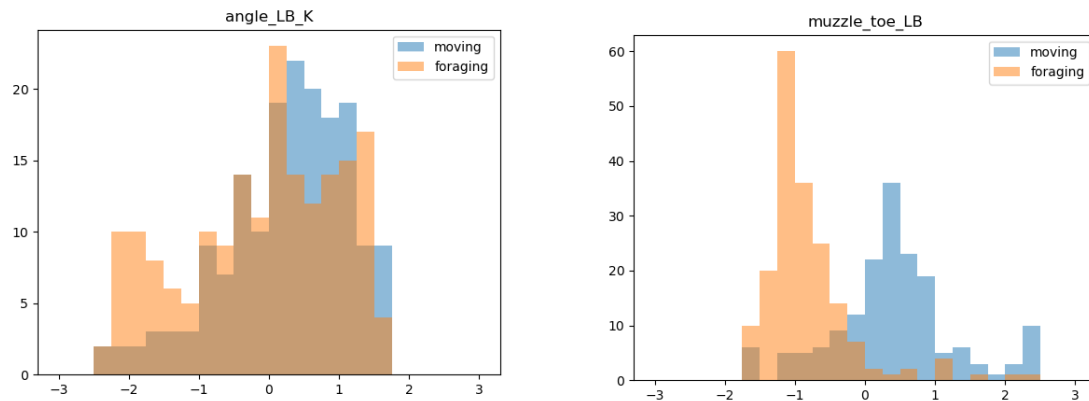


Figure 9 Histograms showing variation in distribution of the classes *moving* (blue) and *foraging* (orange) for the features relating to leg angles (left) and relative distances between head and leg (right)

There are several ways to classify an action based on key-point features. Although it is argued that DL has the best predictive power, there is evidence to suggest that for small datasets a traditional ML method is more reliable (Freytag et al., 2016). Pre-comparison of the DL algorithm, multi layer perceptron (MLP) (with hyperparameters *batch size=32, epochs=1000, hidden layers: 128,256,128*) with a traditional ML RF classifier actually confirmed this hypothesis that the traditional ML RF algorithm on a small dataset performed better than the DL MLP algorithm (Hastie et al., 2009) and therefore RF seemed the more appropriate AR method for PEA. An additional benefit of using the traditional ML algorithm RF was that instead of a black box, a feature importance graph (FIG) and dendrogram could be obtained to give insight in how predictions came about.

The RF algorithm, is an ensemble of decision trees (Hastie et al., 2009). A decision tree algorithm creates multiple binary splits based on optimising homogeneity of the following nodes (Kubat & Kubat, 2017). RF algorithms are usually more accurate than a single decision tree, because each tree contains a random selection of the training data and the final prediction is based on the frequency of class predicted by all the trees (Breiman, 2001). Similarly to YOLOv5, one model was trained for the 3 parent actions and another that included all actions. The key-point features that belonged to the training dataset (which originated from a 23.31px DLC key-point pixel error) were used as input to train the RF recognizing actions. In addition, the key-point features that belonged to the test dataset (which originated from a 217.45px key-point pixel error) were used to test the model's performance of recognized actions.

Training was expedited in seconds and predictions were easily obtained. The predictions were compared with the annotated true action labels by a classification table consisting of accuracy, precision and recall and a confusion matrix. In contrast to the YOLOv5 model that had a confidence threshold for the bounding box prediction, the RF classification algorithm was forced to always choose

the class with the highest confidence, regardless of its confidence value. As YOLOv5 sometimes did not predict any class for challenging images, predictions that were likely for YOLOv5 to be wrong were excluded. Therefore, the YOLOv5 model may deliver more accurate results and appear better than it is. Luckily YOLOv5 only contained 2 false negatives out of 112 test images for the parent actions.

Inference for PEA was more challenging than YOLOv5 because the DLC model that could be applied to new data by the function *analyse time lapse frames* contained the argument `video type = '.avi'` meant for videos instead of individual images. It was not clear how to change this for single images. The RF model on the other hand, was easily downloaded and applied to new data.

2.4.2.3 Box 3. *The history of pose estimation*

A new family of object detection (OD) models, called pose estimation (PE) models, emerged in 2010 outside the area of ecology (Johnson & Everingham, 2010). In PE, the key-points of the posture were estimated and visualized on the animals' body by small dots called key-points. For studying actions this method was a breakthrough because only a few key-points on the animals body could reveal how body parts were related to each other (Johansson, 1973). PE even allowed researchers to do actional predictions based on the pose like *grooming* or tapping for flies (Pereira, Tabris, et al., 2020). Recognition of actions with PE was advantageous over other OD approaches as it only estimated the relevant properties of objects instead of using the pixels with varying value from the object (Mathis et al., 2020). Besides fine-grained control, it was assumed that also less training data was needed to get accurate predictions (Nath et al., 2019).

PE was designed to predict human poses (Z. Cao, Simon, Wei, & Sheikh, 2017; Insafutdinov, Pishchulin, Andres, Andriluka, & Schiele, 2016; Toshev & Szegedy, 2014; Xiao, Wu, & Wei, 2018) and since 2018, has also become available for animals (Pereira, Shaevitz, et al., 2020). For example, PE was used for animals to monitor head action of fish (Huang, He, Wang, & Shen, 2021) and locomotion of cattle (X. Li et al., 2019) and turkeys (Straat, 2020).

Only in recent years pose estimation has been applied on wild animals in captivity, for instance to indoor living Macaques (Bala et al., 2020) and tigers living in national parks (S. Li, Li, Tang, Qian, & Lin, 2019). From a zoo in San Diego it is known that researchers are still struggling with the question as to how PE can contribute to AR (Duhart, Dublon, Mayton, Davenport, & Paradiso, 2019). The pose of a giraffe was predicted as one of the first times that PE was applied to wildlife recorded in their natural environment (Pereira et al., 2019). PE for wildlife is a field with enormous potential however to date, there are no extensive known wildlife studies that have performed PE followed by AR.

2.4.3 Background

2.4.3.2 Box 4. *The history of deep learning*

Early CV classification and OD models mainly relied on part-based models that used parts of images separately to determine the class of an object (Fergus et al., 2003) (Figure 10). For instance, to classify the head of an animal, these models would detect the mouth, the nose and the eye, manually calculate feature distances and from here determine the class by setting manual classification rules. Aside from the fact that it was a time consuming approach, the method consisted mainly of manual steps, in which researchers could create a large bias by making assumptions, about what features were important (Schneider et al., 2019).

ML deals with models that have the ability to learn from the data using algorithms without being explicitly programmed (A. Zhang et al., 2020). This means that ML algorithms can learn recognizing

patterns and use this information to predict new unseen data (Figure 10) (Kotsiantis, Zaharakis, & Pintelas, 2007).

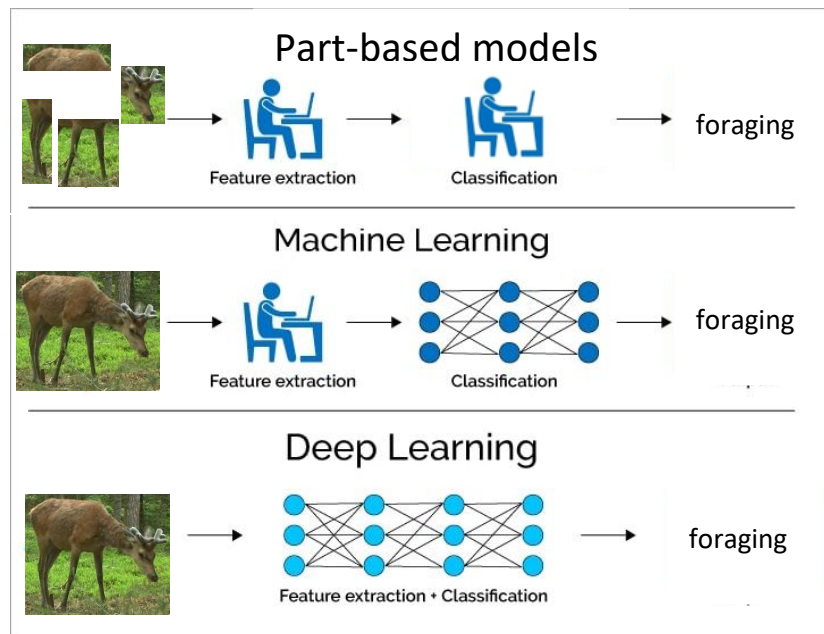


Figure 10 Schematic representation of the history of DL, image source: <https://www.softwaretestinghelp.com/data-mining-vs-machine-learning-vs-ai/>

In supervised ML, like the RF of PEA, many pairs of features and their corresponding labels, are used to train a model, with the features as input and the true label (e.g., *foraging*) as output (Table 5) (Kotsiantis et al., 2007; Norouzzadeh et al., 2018). The model is tested with testing data and hidden labels. To find the accuracy of the model, these hidden labels are retrieved and compared with the models predictions (A. Zhang et al., 2020). ML contains a wide variety of methods from regression to RF and support vector machine (Kubat & Kubat, 2017). For recognizing actions, using ML was advantageous because even the position of obscured body parts could be predicted as it had learned these during training. The pre-determined features however, such as body part distances, that still required ‘hard-coding’ by programmers could still risk bias and were often not applicable to another taxonomic groups (Hiby et al., 2009).

Table 5 Example ML dataset with features (input) and labels (output)

	feature1	feature2	feature3	feature4	label
1	0.21246	-0.98812	-0.09636	3.73765	moving
2	-1.56155	0.58394	-1.24581	-0.64705	foraging
3	0.98039	-0.67589	1.34340	0.80743	other

DL evolved from ML and was different from traditional ML techniques by the multi-layered structure of their neural network that attempted to simulate the action of the human brain (A. Zhang et al., 2020) (Figure 10). In DL even the feature extraction was automated creating an end-to-end framework from image to classification without the need for human interference. These features that the model created automatically could be for example the presence of small or large objects, the position of objects in the image or colour contrast. Despite the abstract concept of automatic feature extraction in which the user did not get to see what the actual features were, automatic feature extraction was a big breakthrough because it eliminated human bias and made vast amounts of information more easily available for ecologists (Norouzzadeh et al., 2018). Due to this advantage, DL often improved

performance over ML methods especially for large datasets (Freytag et al., 2016; Schneider et al., 2019).

Neural networks (NN) exist of multiple layers of which each layer of the builds upon the previous layer to optimize predictions. The most basic form is the multi layered perceptron (MLP) containing, one input, one hidden and one output layer. Even the MLP, can already make approximate predictions (Kratsios, 2021) as activation functions are used that add non-linearity allowing the network to represent any function to fit the data (A. Zhang et al., 2020). An MLP does not work for images but is suitable to recognize actions based on key-point features. Convolutional neural network (CNN) are a class of NN with multiple hidden layers in which for each hidden layer filters are applied to extract information from overlapping regions that turn out to work particularly well for images (Michelucci, 2019; Norouzzadeh et al., 2018). In a CNN, Pixels represented by numerical values are inserted into the network and in each layer, features are abstracted from big details in the first layers like the edge of the back, to more smaller details like an eye, in the last layer (Figure 11). Last, a SoftMax function forces to predict for each action a confidence value between 0 and 1. Depending on the algorithm some like YOLOv5 determines the returned prediction based on a confidence threshold, where other algorithms like RF have the default settings to force choosing prediction even if all confidence values are low. Challenges when training a DL network are that they generally require a lot more input data compared to a ML model because NN consist of many parameters (Michelucci, 2019; Pereira, Shaevitz, et al., 2020).

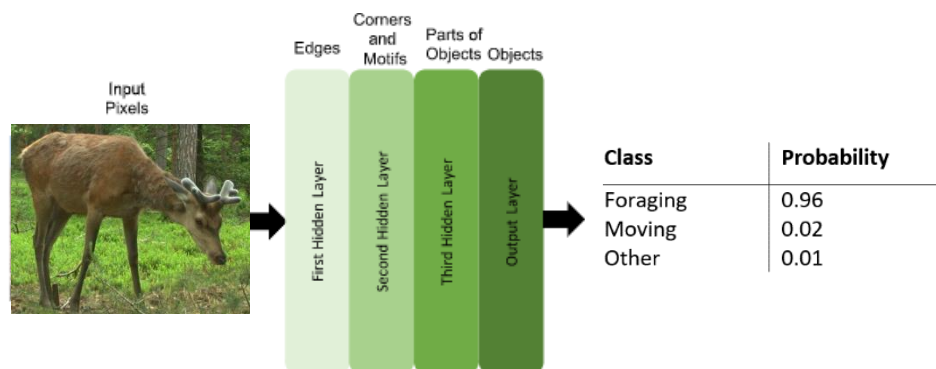


Figure 11 Schematic overview of a convolutional neural network. Learned features are not human-specified but learned by the network (Norouzzadeh et al., 2018).

2.4.3.2 Box 5. Automated action recognition of wildlife

Most DL algorithms have been developed outside the field of ecology in computer image recognition, security or social media industries (Schneider et al., 2019). It is common for developments in camera recognition to shift from controlled and closed set environments like the laboratory to more uncontrolled environments like wildlife (Ravoor & Sudarshan, 2020). Wildlife cameras have to deal with variation in animals viewpoint, image quality, occlusion, dynamic background, weather conditions, seasonality and an unknown number of individuals from different species that might enter and leave the camera at different times (Pereira et al., 2019).

Although criticism that DL for wildlife has not been widely used yet (Schneider et al., 2019), over the last couple of years turtles (Carter, Bell, Miller, & Gash, 2014), primates (Brust et al., 2017; Freytag et al., 2016; Loos & Ernst, 2013) and elephants (Körschens, Barz, & Denzler, 2018) were identified with DL for conservation purposes. AR however can be much more challenging than species identification as actions mostly deal with more small-scale differences (Nath et al., 2019).

During a first attempt to classify wildlife action, in the African savanna system, researchers used the AlexNet model to predict the presence or non-presence of 6 actions. Unfortunately, the classification was based on the entire image and in reviewing the predictions there was a high frequency of predicted actions that were not always clear from the image (Norouzzadeh et al., 2018). Schindler & Steinhage on the other hand used 477 Red deer media files as input to the OD model ResNet, to also classify 3 action classes from Red deer with 63,8% precision (Schindler & Steinhage, 2021).

OD methods that classify wildlife actions based on an animal localisation can prevent overfitting, an advantage that the researchers who used the ResNet OD model likely took into account (Kubat & Kubat, 2017; Michelucci, 2019). To know how YOLOv5 and PEA would perform compared to the method used by Schindler & Steinhage (2001), the same input dataset was required, which unfortunately, was not feasible in this study. Nevertheless, it does seem that a 63.8% precision, which the ResNet OD model achieved, is statistically significant. It is interesting to find out how accurate the YOLOv5 and PEA can predict wildlife actions.

3. Results

For recognition of the parent actions *foraging*, *moving* and *other*, the overall accuracy, which is the percentage of correct predictions by the model, differed between the two approaches (Figure 12). As YOLOv5 contained a confidence threshold, it did not predict 2% of the test dataset images. Of the 98% it predicted, it reached 77% accuracy and for the pose estimation approach (PEA), all images were predicted with an accuracy of 53%.

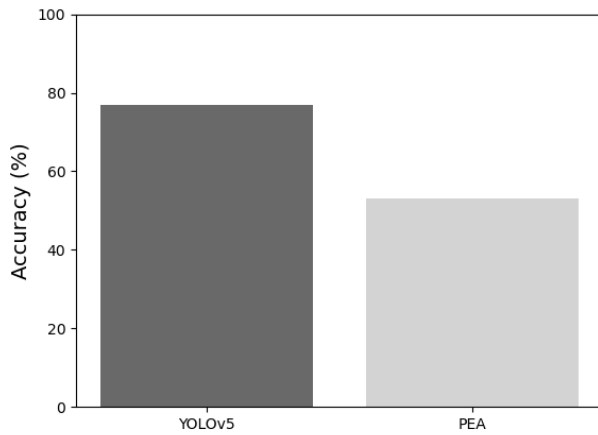


Figure 12 Overall accuracy YOLOv5 and the pose estimation approach (PEA)

The precision, the percentage of a class predictions that is correct (Kubat & Kubat, 2017) was substantially higher for YOLOv5 than for PEA (Figure 13). Both methods had most difficulty predicting the class *other*, probably due to its heterogeneity. Although there were hardly any differences between *foraging* and *moving* for YOLOv5, for PEA, *moving* predictions were significantly worse than *foraging*. Moreover the recall, the percentage of a class that had been correctly classified as that class, was much lower for *moving* (33%) than *foraging* (76%) (Figure 13) (Kubat & Kubat, 2017). It is very likely that, not labelling key-points obscured by the animal has led to erroneous knee key-point predictions which influenced the leg angle features explanatory for *moving* actions. Alternatively, maybe since *moving* related to small scale movement of the legs and *foraging* to larger scale movement of the head PEA had more difficulty detecting small scale details compared to YOLOv5 in which no large differences in precision were observed. YOLOv5's low recall for *moving* (66%), however, showed that the model had difficulty recognizing *moving* images in the dataset.

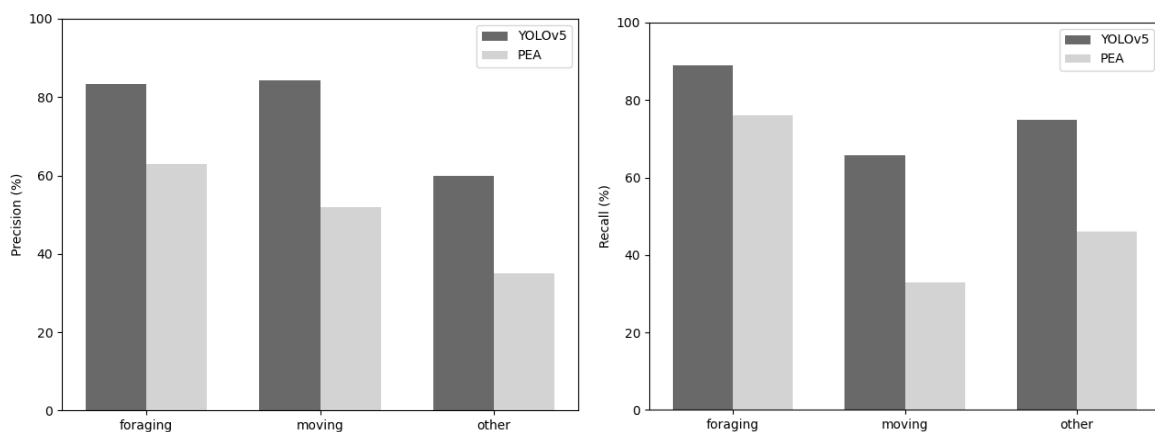


Figure 13 Precision (left) and recall (right) YOLOv5 and the pose estimation approach (PEA)

The confusion matrixes summarize the classes to which the predictions were mostly confused by breaking down the predictions for each class (Figure 14). Ideally, the matrix would have a black diagonal line in which all predicted actions match the true classes. It is noticeable that a significant part of the images YOLOv5 predicted as *other* were, in fact, *moving*, while images predicted as *moving* were rarely predicted as *other*. Again, it is assumed that because *other* was such a heterogeneous group, predicting this class could lead to confusions easily, in contrast to *moving* which mostly consisted of *walking* images. Mismatches in PEA occurred in all classes, which shows the model was confusing all classes with each other.

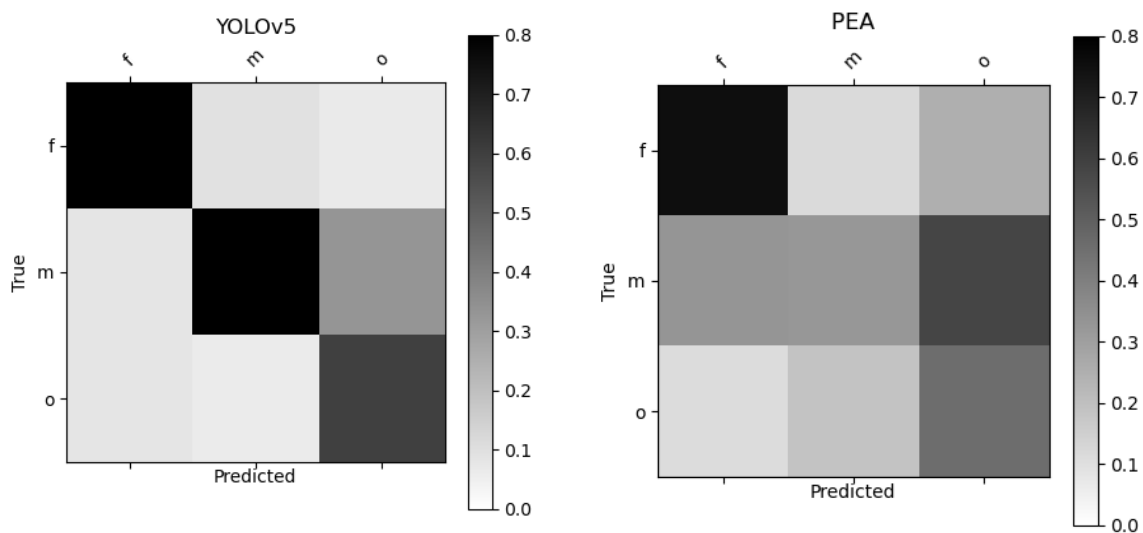


Figure 14 confusion matrix YOLOv5 (left) and the pose estimation approach (PEA) (right). Predicted action (x-axis) and true labelled action (y-axis)

Since in PEA, the RF belonged to the traditional ML methods, a feature importance graph (FIG) was shown indicating which features were most explanatory for AR (Figure 15). It used Mean Decrease in Impurity (MDI) as a measure of how much a splitting criterion gained by including this variable (Breiman, 2001). Although the FIG features contained large error bars, features related to the angles of the legs seemed to be less informative than features dealing with distances between the head and the leg which as mentioned above was not remarkable as these features were strongly influenced by incorrect key-point predictions of key-points occluded by the animal.

Similar findings were found in the dendrogram of a decision tree that used the predictions of the RF as input. This dendrogram showed that head to leg features were used to distinguish *foraging* frames from the classes *moving* and *other*, and that a split by the leg angle feature was not as effective to separate large, homogenized groups (Appendix 8: Results).

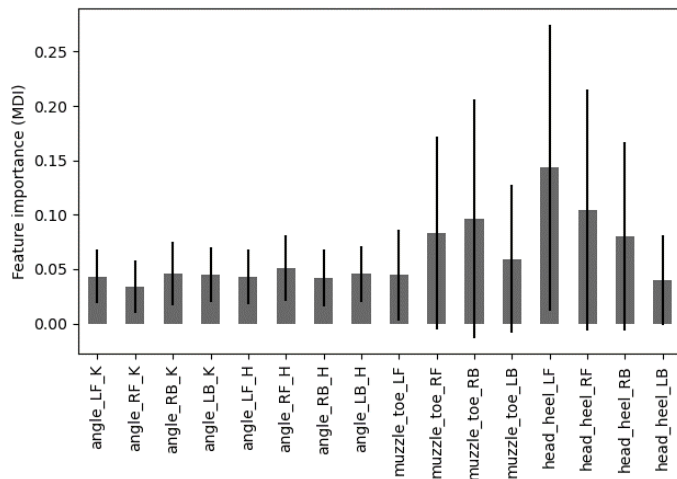


Figure 15 FIG RF (PEA) with mean MDI values \pm SD. 1-8 leg angle features, 9-18 distance head to leg features.

For the models that looked at all 11 actions, YOLOv5 only predicted 92% of the test dataset and only with 50% accuracy (Appendix 8: Results). For PEA all images were predicted but only with an accuracy of 36%. Both methods therefore had considerably more difficulty predicting all behaviours. For the classes with a relatively large number of images, *grazing*, *scanning*, *walking*, *vigilance* and *standing*, most of the time again YOLOv5 performed better than PEA. For the classes with a relatively few images, *browsing*, *running*, *sitting*, *roaring*, *grooming* and *camara watching*, hardly any of these categories were predicted which was not surprising as imbalanced datasets tend to bias towards classes with more examples (Norouzzadeh et al., 2018)

Also looking at the cost in terms of time it took to create, train and apply the models, YOLOv5 outperformed PEA (Table 6). PEA took 35 more hours to create the model mainly due to labelling of key-points, 69 more hours to run the model and extra time to understand how to apply the DLC model to new data. Finally, PEA needed 12,3 GB more GPU memory.

Table 6 Costs of the two modelling approaches for actional classification

	YOLOv5	PEA
Time effort creating model excl. data collection (hrs)	7 (-35)	42 (+35)
Runtime model (hrs)	1 (-69)	DLC 70, RF 0 (+69)
Applying inference (hrs)	1 (+)	>1 (-)
GPU memory (GB)	11.1 (-12,3)	DLC 23.4, RF 0 (+12,3)

4. Discussion

Camera-traps provide snapshots of animals engaged in a variety of actions, which can inform ecology and conservation management (Norouzzadeh et al., 2018), but manual annotation of imagery is time consuming and laborious (Schneider et al., 2019). The aim of this thesis was to find out whether and how ML techniques can automate action recognition (AR). Two methods were compared, YOLOv5 (Bochkovskiy et al., 2020) and the pose estimation approach (PEA) consisting of pose estimation (PE) technique DeepLabCut (DLC) (Mathis et al., 2018) followed by a random forest (RF) algorithm (Kubat & Kubat, 2017). The outcome of the study indicated that YOLOv5 is 15,9% more accurate than PEA and with less effort of creating the models.

YOLOv5 was excellent at predicting *moving* (precision 84%) and *foraging* (precision 83%) images yet the lower recall of *moving* (recall 66%) compared to *foraging* (recall 89%) showed its difficulty detecting *moving* frames in the dataset. Class *other* images were significantly worse predicted (precision 60%) but with a higher recall (recall 75%). As YOLOv5 can achieve a 99,3% average precision with a dataset of at least 400 images per class, it is likely current results can be improved by adding more training data (Liu, Tang, & Zou, 2021; Shahinfar et al., 2020). More training data will be particularly needed when the focus switches from the 3 parent behaviours to all behaviours, where finer details come into play (Bochkovskiy et al., 2020). Comparable AR research in which YouTube videos were used as input to a 3D CNN achieved a much lower accuracy of 36%, but was also more challenging as it was designed to recognize 3 or 4 actions of 32 animal species (W. Li, Swetha, & Shah, 2020).

Looking at the precision and recall PEA performed reasonable predicting class *foraging*, but much worse predicting classes *moving* and *other*. As in PEA precision ranges from 35% to 63%, it is clear that in this study, the supposed benefits for PE such as fine grained control and good performance on small datasets with the current set-up were not found (Nath et al., 2019). The pixel error which is 217.45px yet could be 5px, indicates that a large amount of accuracy has already been lost in DLC (Nath et al., 2019). In related work were, using a neural network, human actions were classified based on pose-key-points, a 90% accuracy was reached which show that it is possible to accurately classify actions form key-points (Song, Zhang, Shan, & Wang, 2020).

YOLOv5 and PEA were compared by accuracy, precision, recall and human effort. Comparison showed YOLOv5 achieved a better performance particularly for action *foraging* with less effort. In comparison to the AR study of Schindler and Steinhage (2021) who obtained 63.8% precision also on classifying 3 Red deer actions, YOLOv5 seems to perform better and PEA much worse, however, it should be noted in this comparison that a different dataset and a segmentation network ResNet in which the exact boundaries of the objects are determined were used (Schindler & Steinhage, 2021).

Extra attention was paid to the prevention of overfitting by choosing OD methods that used animal localisation instead of directly classifying from the full image. Moreover, variation in camera-trap locations, age, habitats, animals shape, animal viewpoint and light conditions was preserved to allow for a better generalization to new data (Beery et al., 2018). Similarly, image augmentation methods mixup and imgaug that artificially increased training data, have likely contributed to preventing overfitting by adding complexity to the training process and lowering the error on the test dataset (Michelucci, 2019). Particularly the good results of YOLOv5s class *foraging* shows that the model has been able to predict new data correctly. Further research would be needed to better understand how the individual overfitting measures have contributed to a better generalization.

A limitation to YOLOv5 was that 2 images did not contain any prediction while these were initially labelled with a Red deer and thus, these were classified as false negatives, causing analysis on the number of wrong predictions turned out slightly more advantageous for YOLOv5 than it was. YOLOv5 could be improved by increasing the dataset and testing different hyperparameter values like number of epochs and batch size. Although these measures are also expected to improve PEA, for PEA, there were also several other presumed causes for the lower performance in PEA.

First, DLC was trained on full images and with a MAE 217.45px was likely to have resulted in overfitting by background disturbance (Michelucci, 2019). Second, in hindsight not annotating key-points obscured by the animal was a handicap, as these key-points were often predicted on other body parts affecting key-point features especially related to the leg angle important for *moving* actions. Third, while the RF algorithm outperformed alternatives such as MLP, maybe an alternative DL action classifier with optimized hyperparameters is more effective.

A limitation to this study in general was optimal hyperparameter values were obtained by running methods several times, making the network good at predicting the test dataset but less for predicting new data. Another limitation was that due to the predetermined train and test split, the results of this study have not been validated by cross validation, a resampling method using different proportions of the data for training and testing on different iterations, which can show results are not out of "luck", but constantly performs around a certain accuracy (Kubat & Kubat, 2017).

As DL for wildlife AR had previously not been extensively explored (W. Li et al., 2020), this research could bridge the gap and bring major innovations. Conservationists now can easily explore large volumes of action data for Red deer (*Cervus elaphus*) or any other animal so that they can obtain a more complete picture of wildlife behaviour. While method PEA still needs improvement, conservationists can directly apply the robust YOLOv5 model for instance to determine feeding behaviour of wild livestock (Oliveira, Pereira, Bresolin, Ferreira, & Dorea, 2021), recognize migration patterns in response to climate change (Davidson et al., 2020), detect abnormalities in responses to poachers (de Knecht, Eikelboom, van Langevelde, Spruyt, & Prins, 2021) or estimate daily activity patterns (Rowcliffe, Kays, Kranstauber, Carbone, & Jansen, 2014) and determine how these differ from the global activity pattern (Hester, Gordon, Baillie, & Tappin, 1999; Jayakody, Sibbald, Gordon, & Lambin, 2008; Rattray, 2009).

Based on the recent findings, it may still be worth doing PEA again, but it makes sense to use the animals bounding box as input for DLC, always annotate all visible and obscured key-points, and add additional key-point features especially when expanding to all actions. An additional advantage of taking the bounding box, which is present in YOLOv5, is that classification per bounding box is possible and can therefore apply AR to an image with several animals. Moreover it is recommended that DL alternatives be explored for AR such as the graphical convolutional neural network ResGCN, which also includes a key-point skeleton and can classify actions based on consecutive images (Song et al., 2020). It would be interesting to deduce whether YOLOv5 is still better when looking at consecutive images as it maybe YOLOv5 just provides a classification for each individual image, whereas ResGCN includes information on key-point changes from image to image (H. Cao et al., 2021).

In general, to prove that the results are statistically sound it is recommended to add cross validation to both methods. To reduce overfitting, it is recommended to add a validation set used to find optimal values for the hyperparameters which can then be applied to the test dataset (Van Etten, Lindenbaum, & Bacastow, 2018). To increase performance is highly recommended, whether the aim is to classify the parent actions or all actions, to use at least 500 images per class to obtain a good accuracy (Norouzzadeh et al., 2018; Shahinfar et al., 2020). Action labelling effort can be shortened by

implementing the functionality of action annotation for species into annotation platforms so that researchers can collectively contribute to a database of action annotated training data. Expanding to reliable models for all behaviours will be manageable as actions and extensive key-point labels are already annotated, only requiring expanding the dataset and refining the key-point features. Both methods can be easily applied to recognize behaviour for other animals, only requiring refining actions and animals key-points.

5. Reference list

- Altmann, J. (1974). Observational study of behavior: sampling methods. *Behaviour*, 49(3-4), 227-266.
- Bala, P. C., Eisenreich, B. R., Yoo, S. B. M., Hayden, B. Y., Park, H. S., & Zimmermann, J. (2020). Automated markerless pose estimation in freely moving macaques with OpenMonkeyStudio. *Nature communications*, 11(1), 1-12.
- Beery, S., Van Horn, G., & Perona, P. (2018). *Recognition in terra incognita*. Paper presented at the Proceedings of the European conference on computer vision (ECCV).
- Bochkovskiy, A., Wang, C.-Y., & Liao, H.-Y. M. (2020). Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*.
- Braude, S., Margulis, S., & Broder, E. D. (2017). The Study of Animal Behavior Provides Valuable Opportunities for Original Science Fair Projects: Recommendations from The Animal Behavior Society, Education Committee. *The American Biology Teacher*, 79(6), 438-441.
- Breiman, L. (2001). Random forests. *machine learning*, 45(1), 5-32.
- Brust, C.-A., Burghardt, T., Groenenberg, M., Kading, C., Kuhl, H. S., Manguette, M. L., & Denzler, J. (2017). *Towards automated visual monitoring of individual gorillas in the wild*. Paper presented at the Proceedings of the IEEE International Conference on CV Workshops.
- Bunkley, N. (2009). Joseph Juran," Pioneer in Quality Control, Dies. *New York Times*, 103.
- Camtrape DP Development Team. (2021). Camera Trap Data Package. Retrieved from <https://tdwg.github.io/camtrap-dp/>
- Cao, H., Xu, Y., Yang, J., Mao, K., Yin, J., & See, S. (2021). Effective action recognition with embedded key point shifts. *Pattern Recognition*, 120, 108172.
- Cao, Z., Simon, T., Wei, S.-E., & Sheikh, Y. (2017). *Realtime multi-person 2d pose estimation using part affinity fields*. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
- Carter, S. J., Bell, I. P., Miller, J. J., & Gash, P. P. (2014). Automated marine turtle photograph identification using artificial neural networks, with application to green turtles. *Journal of experimental marine biology and ecology*, 452, 105-110.
- Casaer, J., Milotic, T., Liefting, Y., Desmet, P., & Jansen, P. (2019). Agouti: A platform for processing and archiving of camera trap images. *Biodiversity Information Science and Standards*.
- Davidson, S. C., Bohrer, G., Gurarie, E., LaPoint, S., Mahoney, P. J., Boelman, N. T., . . . Jennewein, J. (2020). Ecological insights from three decades of animal movement tracking across a changing Arctic. *Science*, 370(6517), 712-715.
- de Knecht, H. J., Eikelboom, J. A., van Langevelde, F., Spruyt, W. F., & Prins, H. H. (2021). Timely poacher detection and localization using sentinel animal movement. *Scientific reports*, 11(1), 1-11.
- Duhart, C., Dublon, G., Mayton, B., Davenport, G., & Paradiso, J. A. (2019). *Deep learning for wildlife conservation and restoration efforts*. Paper presented at the 36th International Conference on machine learning, Long Beach.
- Fergus, R., Perona, P., & Zisserman, A. (2003). *Object class recognition by unsupervised scale-invariant learning*. Paper presented at the 2003 IEEE Computer Society Conference on computer vision and Pattern Recognition, 2003. Proceedings.
- Freytag, A., Rodner, E., Simon, M., Loos, A., Kuhl, H. S., & Denzler, J. (2016). *Chimpanzee faces in the wild: Log-euclidean CNNs for predicting identities and attributes of primates*. Paper presented at the German Conference on Pattern Recognition.
- Gebert, C., & Verheyden-Tixier, H. (2001). Variations of diet composition of red deer (*Cervus elaphus* L.) in Europe. *Mammal Review*, 31(3-4), 189-201.
- Gils, J. v. (2021). Behaviour annotation protocol for camera-trap data in R. Retrieved from <https://rpubs.com/JorritvanGils/BehaviourProtocolR>
- Girshick, R. (2015). *Fast r-cnn*. Paper presented at the Proceedings of the IEEE international conference on computer vision.

- Graving, J. M., Chae, D., Naik, H., Li, L., Koger, B., Costelloe, B. R., & Couzin, I. D. (2019). DeepPoseKit, a software toolkit for fast and robust animal pose estimation using deep learning. *Elife*, *8*, e47994.
- Graystock, P., & Hughes, W. O. (2011). Disease resistance in a weaver ant, *Polyrhachis dives*, and the role of antibiotic-producing glands. *Behavioral Ecology and Sociobiology*, *65*(12), 2319-2327.
- Grieser, J., Gommers, R., Cofield, S., & Bernardi, M. (2006). New gridded maps of Koeppen's climate classification. *Data, methodology and gridded data* Available at: http://www.fao.org/nr/climpag/globgrids/KC_classification_en.asp. Methodology also downloadable from http://www.juergen-grieser.de/downloads/Koeppen-Climatology/Koeppen_Climatology.pdf (accessed September 2015).
- Hastie, T., Tibshirani, R., Friedman, J. H., & Friedman, J. H. (2009). *The elements of statistical learning: data mining, inference, and prediction* (Vol. 2): Springer.
- Hester, A., Gordon, I., Baillie, G., & Tappin, E. (1999). Foraging behaviour of sheep and red deer within natural heather/grass mosaics. *Journal of Applied Ecology*, *36*(1), 133-146.
- Hiby, L., Lovell, P., Patil, N., Kumar, N. S., Gopalaswamy, A. M., & Karanth, K. U. (2009). A tiger cannot change its stripes: using a three-dimensional model to match images of living tigers and tiger skins. *Biology letters*, *5*(3), 383-386.
- Huang, Z.-J., He, X.-X., Wang, F.-J., & Shen, Q. (2021). A Real-Time Multi-Stage Architecture for Pose Estimation of Zebrafish Head with Convolutional Neural Networks. *Journal of Computer Science and Technology*, *36*(2), 434-444.
- Insafutdinov, E., Pishchulin, L., Andres, B., Andriluka, M., & Schiele, B. (2016). *Deepcut: A deeper, stronger, and faster multi-person pose estimation model*. Paper presented at the European Conference on CV.
- Janis, K. M. (1998). *Evolution of tertiary mammals of North America: Volume 1, terrestrial carnivores, ungulates, and ungulate like mammals* (Vol. 1): Cambridge University Press.
- Jayakody, S., Sibbald, A. M., Gordon, I. J., & Lambin, X. (2008). Red deer *Cervus elephus* vigilance behaviour differs with habitat and type of human disturbance. *Wildlife biology*, *14*(1), 81-91.
- Johansson, G. (1973). Visual perception of biological motion and a model for its analysis. *Perception & psychophysics*, *14*(2), 201-211.
- Johnson, S., & Everingham, M. (2010). *Clustered Pose and Nonlinear Appearance Models for Human Pose Estimation*. Paper presented at the bmvc.
- Körschens, M., Barz, B., & Denzler, J. (2018). Towards automatic identification of elephants in the wild. *arXiv preprint arXiv:1812.04418*.
- Kotsiantis, S. B., Zaharakis, I., & Pintelas, P. (2007). Supervised machine learning: A review of classification techniques. *Emerging artificial intelligence applications in computer engineering*, *160*(1), 3-24.
- Kratsios, A. (2021). The universal approximation property. *Annals of Mathematics and Artificial Intelligence*, *89*(5), 435-469.
- Kubat, M., & Kubat. (2017). *An introduction to machine learning* (Vol. 2): Springer.
- Lagardère, J.-P., Anras, M.-L. B., & Claireaux, G. (1999). *Advances in invertebrates and fish telemetry*. Paper presented at the Annales de limnologie.
- Lehner, P. N. (1992). Sampling methods in behavior research. *Poultry science*, *71*(4), 643-649.
- Li, S., Li, J., Tang, H., Qian, R., & Lin, W. (2019). ATRW: a benchmark for Amur tiger re-identification in the wild. *arXiv preprint arXiv:1906.05586*.
- Li, W., Swetha, S., & Shah, M. (2020). Wildlife action recognition using deep learning.
- Li, X., Cai, C., Zhang, R., Ju, L., & He, J. (2019). Deep cascaded convolutional models for cattle pose estimation. *Computers and Electronics in Agriculture*, *164*, 104885.
- Lin, T.-Y., Maire, M., Belongie, S., Hays, J., Perona, P., Ramanan, D., . . . Zitnick, C. L. (2014). *Microsoft coco: Common objects in context*. Paper presented at the European conference on computer vision.

- Liu, X., Tang, G., & Zou, W. (2021). *Improvement of Detection Accuracy of Aircraft in Remote Sensing Images Based on YOLOV5 Model*. Paper presented at the 2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS.
- Loos, A., & Ernst, A. (2013). An automated chimpanzee identification system using face detection and recognition. *EURASIP Journal on Image and Video Processing*, 2013(1), 1-17.
- Mathis, A., Mamidanna, P., Cury, K. M., Abe, T., Murthy, V. N., Mathis, M. W., & Bethge, M. (2018). DeepLabCut: markerless pose estimation of user-defined body parts with deep learning. *Nature neuroscience*, 21(9), 1281-1289.
- Mathis, A., Schneider, S., Lauer, J., & Mathis, M. W. (2020). A primer on motion capture with deep learning: principles, pitfalls, and perspectives. *Neuron*, 108(1), 44-65.
- Mellor, D. J., Beausoleil, N. J., & Stafford, K. J. (2004). *Marking amphibians, reptiles and marine mammals: animal welfare, practicalities and public perceptions in New Zealand*: Department of Conservation Wellington.
- Mench, J. (1998). Why it is important to understand animal behavior. *ILAR journal*, 39(1), 20-26.
- Michelucci, U. (2019). *Advanced applied deep learning: convolutional neural networks and object detection*: Springer.
- Nath, T., Mathis, A., Chen, A. C., Patel, A., Bethge, M., & Mathis, M. W. (2019). Using DeepLabCut for 3D markerless pose estimation across species and behaviors. *Nature protocols*, 14(7), 2152-2176.
- National Park Hoge Veluwe. (2021). Red deer, the king of the park. Retrieved from <https://www.hogeveluwe.nl/en/discover-the-park/nature-and-landscape/red-deer>
- Norouzzadeh, M. S., Nguyen, A., Kosmala, M., Swanson, A., Palmer, M. S., Packer, C., & Clune, J. (2018). Automatically identifying, counting, and describing wild animals in camera-trap images with deep learning. *Proceedings of the National Academy of Sciences*, 115(25), E5716-E5725.
- Oliveira, D. A. B., Pereira, L. G. R., Bresolin, T., Ferreira, R. E. P., & Dorea, J. R. R. (2021). A review of deep learning algorithms for computer vision systems in livestock. *Livestock Science*, 253, 104700.
- Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., . . . Dubourg, V. (2011). Scikit-learn: machine learning in Python. *the Journal of machine learning research*, 12, 2825-2830.
- Pereira, T. D., Aldarondo, D. E., Willmore, L., Kislin, M., Wang, S. S.-H., Murthy, M., & Shaevitz, J. W. (2019). Fast animal pose estimation using deep neural networks. *Nature methods*, 16(1), 117-125.
- Pereira, T. D., Shaevitz, J. W., & Murthy, M. (2020). Quantifying behavior to understand the brain. *Nature neuroscience*, 23(12), 1537-1549.
- Pereira, T. D., Tabris, N., Li, J., Ravindranath, S., Papadoyannis, E. S., Wang, Z. Y., . . . Falkner, A. L. (2020). SLEAP: multi-animal pose tracking. *bioRxiv*.
- R Core Team. (2020). R: A language and environment for statistical computing. R Foundation for Statistical Computing. Vienna, Austria. Retrieved from <https://www.R-project.org/>
- Rattray, P. (2009). *Red Deer Hunting: A Complete Guide*: Paul Rattray.
- Ravoor, P. C., & Sudarshan, T. (2020). Deep learning Methods for Multi-Species Animal Re-identification and Tracking—a Survey. *Computer Science Review*, 38, 100289.
- Redmon, J., Divvala, S., Girshick, R., & Farhadi, A. (2016). *You only look once: Unified, real-time object detection*. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
- Rowcliffe, J. M., Kays, R., Kranstauber, B., Carbone, C., & Jansen, P. A. (2014). Quantifying levels of animal activity using camera trap data. *Methods in Ecology and Evolution*, 5(11), 1170-1179.
- Schindler, F., & Steinhage, V. (2021). Identification of animals and recognition of their actions in wildlife videos using deep learning techniques. *Ecological Informatics*, 61, 101215.

- Schneider, S., Taylor, G. W., Linquist, S., & Kremer, S. C. (2019). Past, present and future approaches using computer vision for animal re-identification from camera trap data. *Methods in Ecology and Evolution*, 10(4), 461-470.
- Shahinfar, S., Meek, P., & Falzon, G. (2020). "How many images do I need?" Understanding how sample size per class affects deep learning model performance metrics for balanced designs in autonomous wildlife monitoring. *Ecological Informatics*, 57, 101085.
- Song, Y.-F., Zhang, Z., Shan, C., & Wang, L. (2020). *Stronger, faster and more explainable: A graph convolutional baseline for skeleton-based action recognition*. Paper presented at the proceedings of the 28th ACM international conference on multimedia.
- Straat, T. v. d. (2020). *Efficient measuring of locomotion of farm animals*. Retrieved from
- Tan, M., Pang, R., & Le, Q. V. (2020). *Efficientdet: Scalable and efficient object detection*. Paper presented at the Proceedings of the IEEE/CVF conference on computer vision and pattern recognition.
- Toshev, A., & Szegedy, C. (2014). *Deeppose: Human pose estimation via deep neural networks*. Paper presented at the Proceedings of the IEEE conference on computer vision and pattern recognition.
- Tuia, D., Kellenberger, B., Beery, S., Costelloe, B. R., Zuffi, S., Risse, B., . . . Burghardt, T. (2022). Perspectives in machine learning for wildlife conservation. *Nature communications*, 13(1), 1-15.
- Van Etten, A., Lindenbaum, D., & Bacastow, T. M. (2018). Spacenet: A remote sensing dataset and challenge series. *arXiv preprint arXiv:1807.01232*.
- Van Rossum, G. (2020). The Python Library Reference, release 3.8. 2. *Python Software Foundation*, 36.
- Van Rossum, G., & Drake Jr, F. L. (1995). *Python tutorial* (Vol. 620): Centrum voor Wiskunde en Informatica Amsterdam, The Netherlands.
- Wada. (2022). Mammal's Locomotion. Retrieved from <http://mammals-locomotion.com/walking.html>
- Waldrop, M. M. (2019). News Feature: What are the limits of deep learning? *Proceedings of the National Academy of Sciences*, 116(4), 1074-1077.
- Xiao, B., Wu, H., & Wei, Y. (2018). *Simple baselines for human pose estimation and tracking*. Paper presented at the Proceedings of the European conference on computer vision (ECCV).
- Zhang, A., Lipton, Z. C., Li, M., & Smola, A. J. (2020). Dive into deep learning. 2020. URL <https://d2l.ai>.
- Zhang, H., Cisse, M., Dauphin, Y. N., & Lopez-Paz, D. (2017). mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*.

Appendix 1: Map National Park Hoge Veluwe

NP De Hoge Veluwe

Camera Network Final
Stratification V3

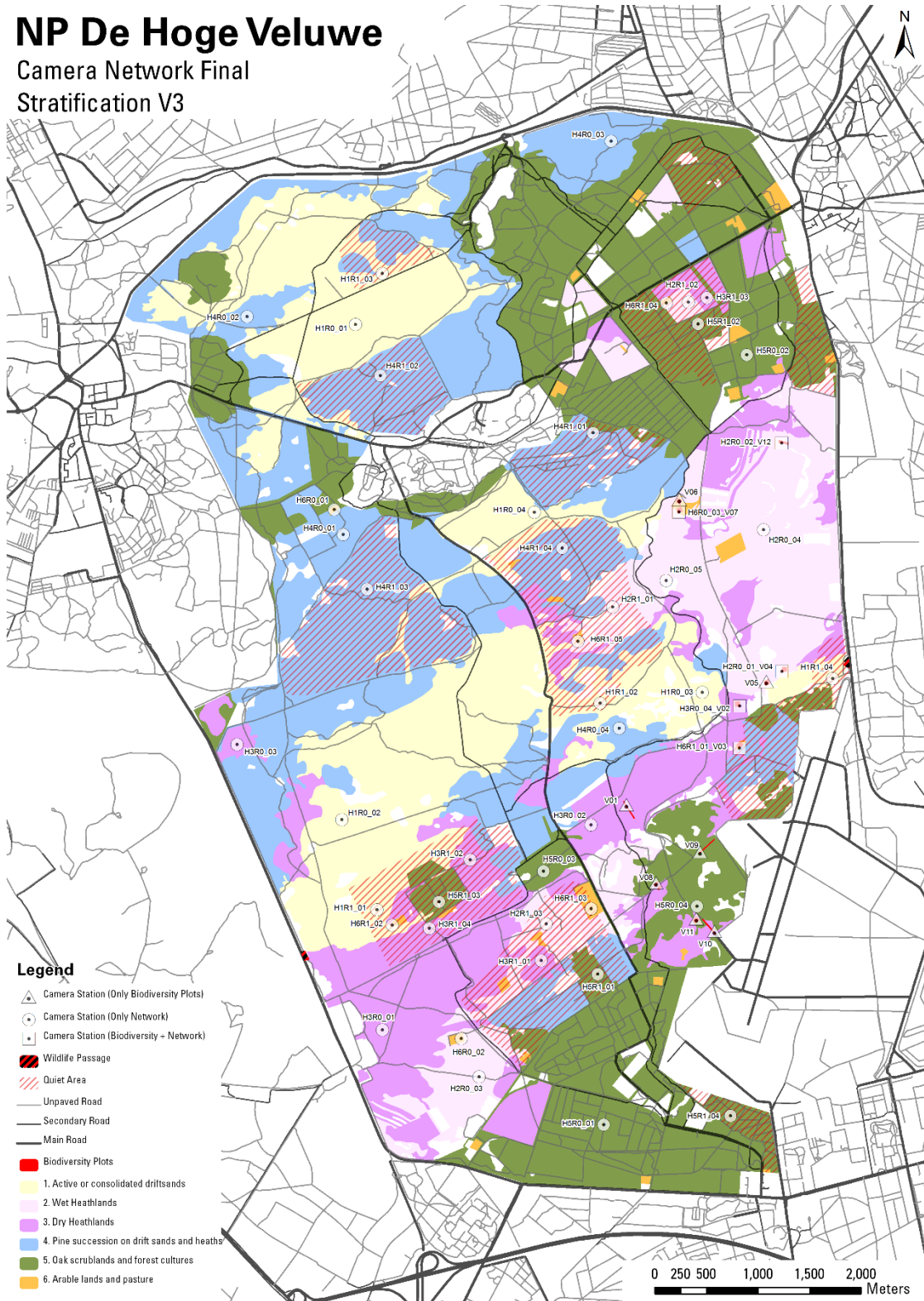













Figure 16 Detailed map of NPHV including camera locations

Appendix 2: Action annotation protocol

The Red deer AR criteria are shown (Table 7). The protocol for annotating wildlife actions from camera trap images is included in the reference list (Gils, 2021).

Table 7 Red deer AR criteria

<p>1.Grazing</p>  <p>Mouth in touch with ground or grass vegetation</p>	<p>2.Browsing</p>  <p>Mouth in contact with woody vegetation. Stable leg position.</p>	<p>3.Scanning</p>  <p>Mouth reaches up or down but does not touch the vegetation. Can occur either while <i>moving</i> or <i>standing</i>.</p>	<p>4.Walking</p>  <p>Diagonal walk pattern. Front foot initiates and diagonal hind leg follows. Always two contact points, generally 3.</p>
<p>5.Running</p>  <p>Aerial phase with max 2 contact points visible. include trot, canter and gallop</p>	<p>6.Sitting</p>  <p>Sitting or laying on the ground.</p>	<p>7.Standing</p>  <p>Legs generally parallel. Not particularly tense leg position and low ear position.</p>	<p>8.Vigilance</p>  <p>Tense leg position in combination with pointed upward ears</p>
<p>9.Grooming</p>  <p>Snout to body or leg, leg to body part or body part to object</p>	<p>10.Roaring</p>  <p>Head faced upwards and mouth opens</p>	<p>11.Camera watching</p>  <p>Face and eyes point towards the camera</p>	

Appendix 3: Scripts and models

A. Dataset creation

- A01_data_selection.R
- A02_download_and_image_selection.R
- A03_collect_and_preprocess.R
- A04_labeling.R
- A05_create_labels_table.R
- A06_exploring_data.R

B. YOLOv5

- B01_box21_Reddeer.R
- B02_object_detection_inference.py
- B03_confusion_matrix.py

C. Pose estimation approach

- C01_terminal_DLC.py
- C02_GPU_PC_commands
- C03_create_training_dataset.py
- C04_train_and_evaluate_network.py
- C05_padding.py
- C06_analyse_time_lapse_images.py
- C07_feature_extraction.py
- C08_outliers_and_scaling.py
- C09_random_forest.py
- C10_MLP.py
- C11_dendrogram.py

D. General

- D01_Graphs_results.py

E. Models

- E01_YOLOv5_model_parent_behaviour.pt
- E02_YOLOv5_model_behaviour.pt
- E03_PEA_RF_model_parent.joblib
- E04_PEA_RF_model.joblib

Appendix 4: BOX21 example input

	A	B	C	D	E
1		path	original_category	meta	in_validation_set
2	1	https://multimedia.agouti.eu/assets/00add8fd-bd07-4d43-a11d-ae8c30808c7e/file	m	{"seq_id":	FALSE

Figure 17 Example csv input for the YOLOV5 model on BOX21

Table 8 Description of information required per column

Column	Description
path	https://multimedia.agouti.eu/assets/00add8fd-bd07-4d43-a11d-ae8c30808c7e/file
original_class	m
meta	("seq_id": "46acc55c-0b90-41dd-bda7-e373b5d8651c", "depl_id": "2cc8d313-ba9c-43c6-b5d0-f254f906f2d5", "filename": "00add8fd-bd07-4d43-a11d-ae8c30808c7e.JPG") *
in_validation_set	FALSE

***Remark: change the brackets () from column meta to curly brackets.**

Appendix 5: Train and test error graphs

Learning of the model is shown by evaluating the train and test error during training of the model. For YOLOv5 the train and test error are described by the related train and test loss (Figure 18), which describes the penalty for bad predictions, where a low loss indicates the model can predict well (Hastie et al., 2009). Results show that the loss on predicting bounding boxes is low, but prediction for the classes on the test (val) set is much higher, especially for all actions. The PEA DLC error shows how during training the average key-point pixel error evolves for the training and test dataset separately (Figure 19). Training PEA RF took little time, and no loss or train test error was printed

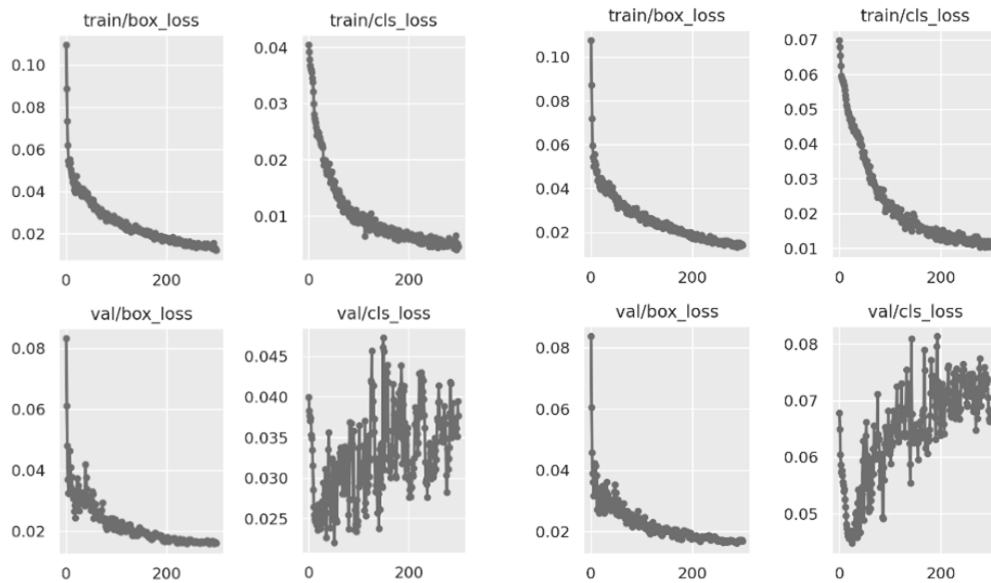


Figure 18 box loss and class loss for the parent actions (left) and all actions (right)

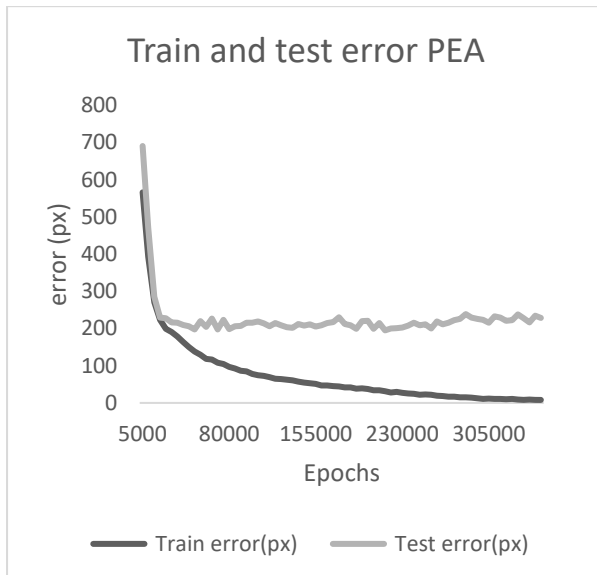


Figure 19 DLC train and test error

Appendix 6: DLC annotation protocol

The DLC annotation protocol is created to provide sufficient and accurate information about the position of the animal's key-points

To label the key-points consistently pre-decisions were made about which key-points were annotated. Based on skeleton analysis, real life camera trap images and an key-point annotated horse example (Nath et al., 2019), 18 key-points were defined covering the important Red deer key-points, while still being visible from the outside (Figure 20). To obtain optimal label accuracy annotating the 506 images was divided into 25-minute parts of annotation alternated by 5 minutes break with a maximum of 3 annotation hours per day.

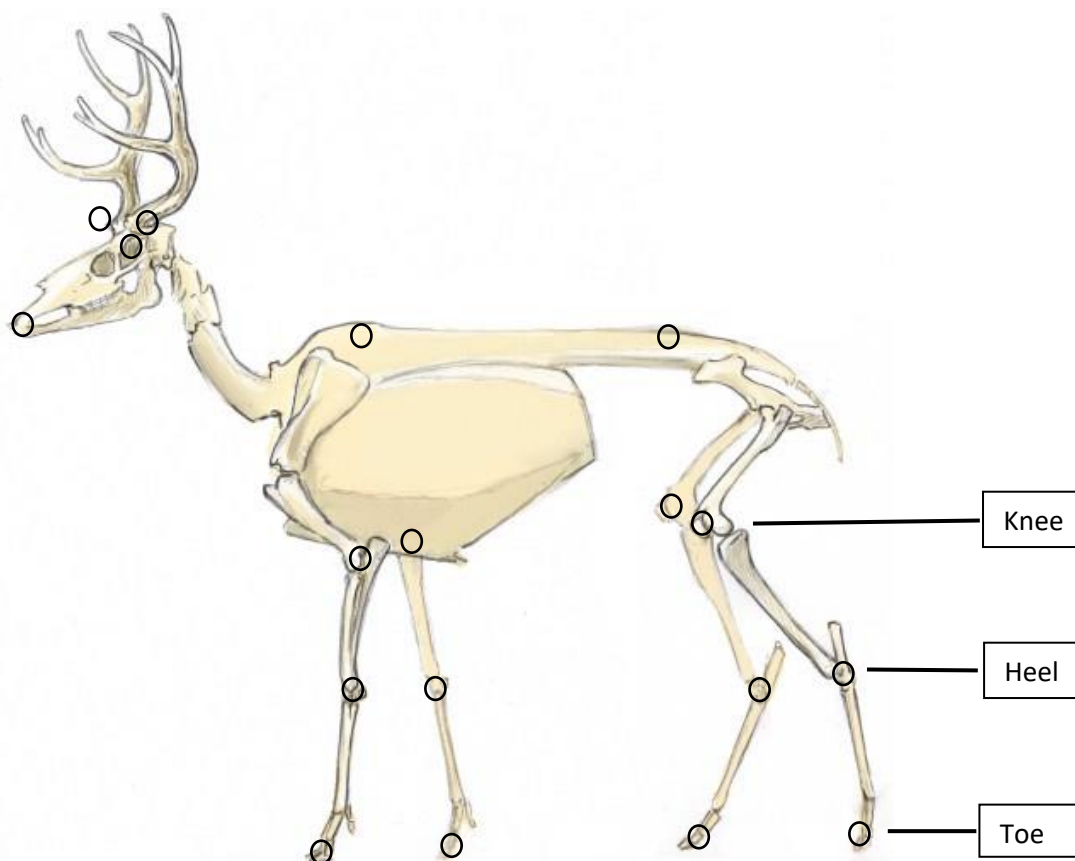


Figure 20 Key-point representation on skeleton. Image from <https://johnmurlaws.com/draw-deer-anatomy/>

The key-points have been chosen relevant for detect all the 11 Red deer actions.

Not all key-points are annotated in every image. An overview of the annotation criteria is shown (Table 9). Visible key-points are annotated and obscured key-points only if occlusion is caused by non-animal objects.

Table 9 annotation criteria

Scenario	Occlusion	Annotated/Predicted
Visible key-points	no	Yes
Obscured key-points	Yes, by non-animal objects (e.g.: vegetation)	Yes
Obscured key-points	Yes, by the animal itself	No

Annotation examples show key-point annotation from the side (Figure 21) and the back (Figure 22). Key-points that were obscured by the animals themselves, were not annotated, to reduce guesswork and thereby reduce the likelihood of negatively influencing the key-point prediction. As key-points obscured by non-animal objects like vegetation are often easy to deduce, these key-points were annotated. To stimulate the neural network detecting contrast with the background, apparent key-points locations were chosen like the black contrasting muzzle or at the tip of the ear that contrasts with its background.

In the example below the code of a key-point example is explained.

Example: L_leg_Fw_T
 1_2__3__4

1. = (L)eft or (R)ight
2. = leg or ear
3. = (F)or(w)ard or (B)ack
4. = (T)oe

To label images consistently, an order is predefined always annotating from toe to heel to knee and from left front leg to right front to right back to left back followed by the back and finally the head key-points (Table 10).

Table 10 Key-point label names

Key-point labels			
1. L_leg_Fw_T	2. L_leg_Fw_K	3. L_leg_Fw_B	4. R_leg_Fw_T
5. R_leg_Fw_K	6. R_leg_Fw_B	7. R_leg_B_T	8. R_leg_B_K
9. R_leg_B_B	10. L_leg_B_T	11. L_leg_B_K	12. L_leg_B_B
13. Low_back	14. High_back	15. Top_head	16. L_ear
17. R_ear	18. Muzzle		

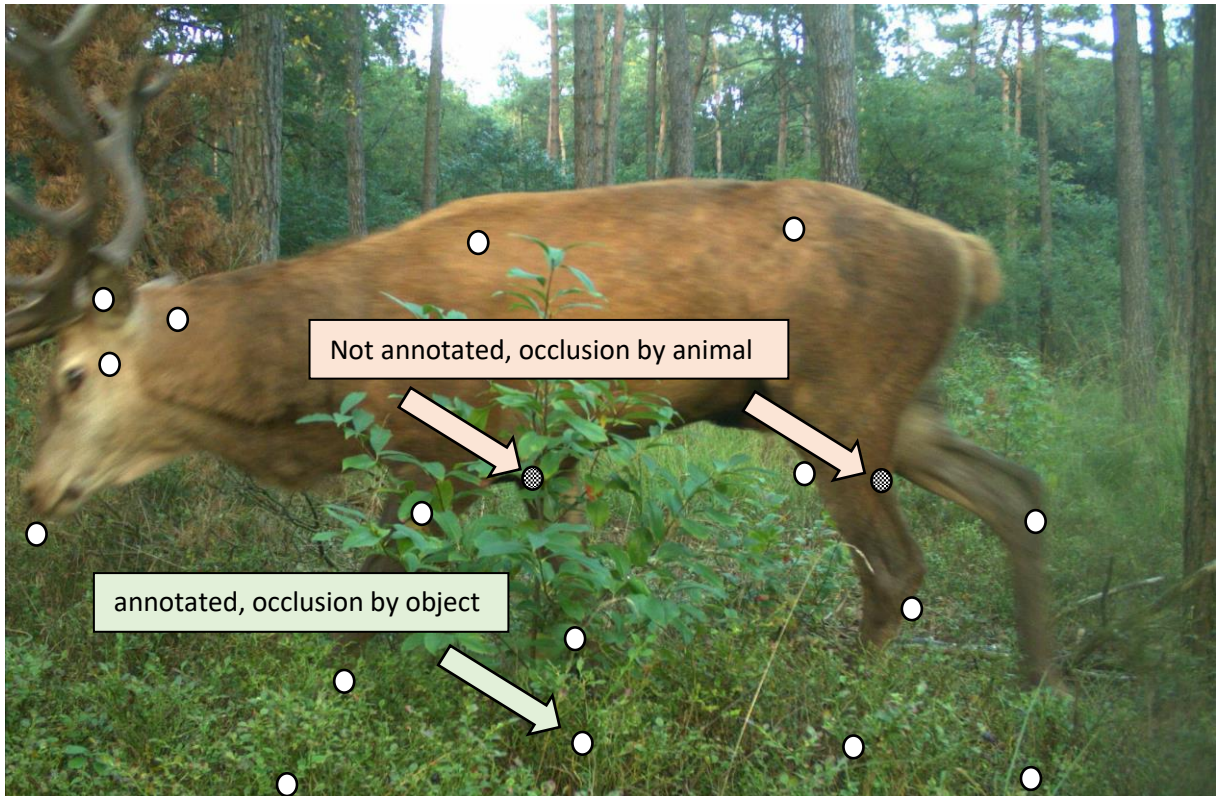


Figure 21 annotation example sideview

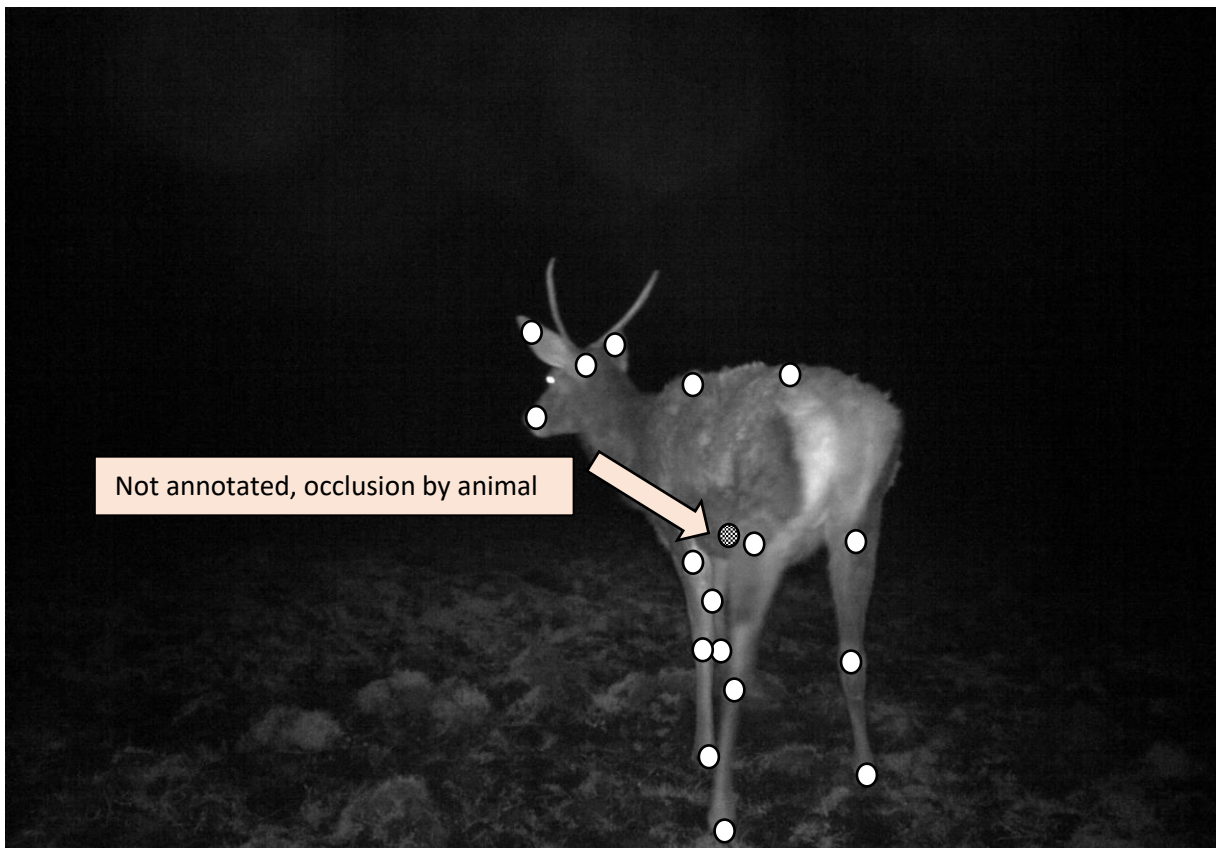


Figure 22 Annotation example behind view

Appendix 7: PEA key-point features and distribution

Table 11 Feature description of the feature extraction

Nr.	feature	abbreviation
1	shortest angle of the heel with the toe and the knee	angle_K
2	shortest angle of the knee with heel, and a hypothetical point with x value of heel and y value of knee.	angle_H
3	Vertical distance muzzle to toe, relative to vertical distance heel to toe	muzzle_toe
4	Vertical distance head to heel, relative to vertical distance knee to heel	head_heel

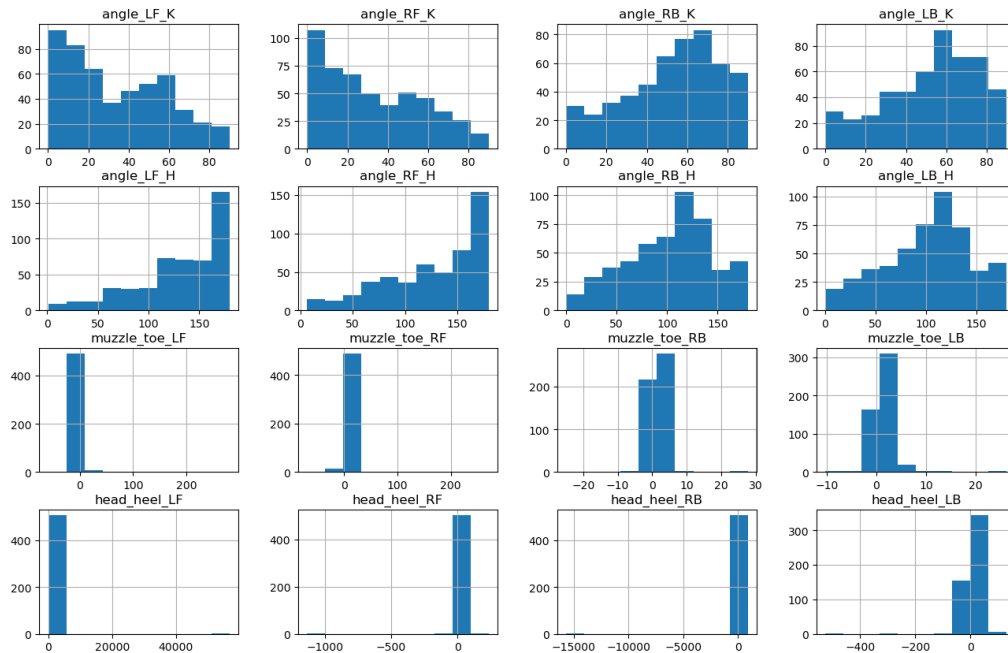


Figure 23 Distribution of all features before scaling

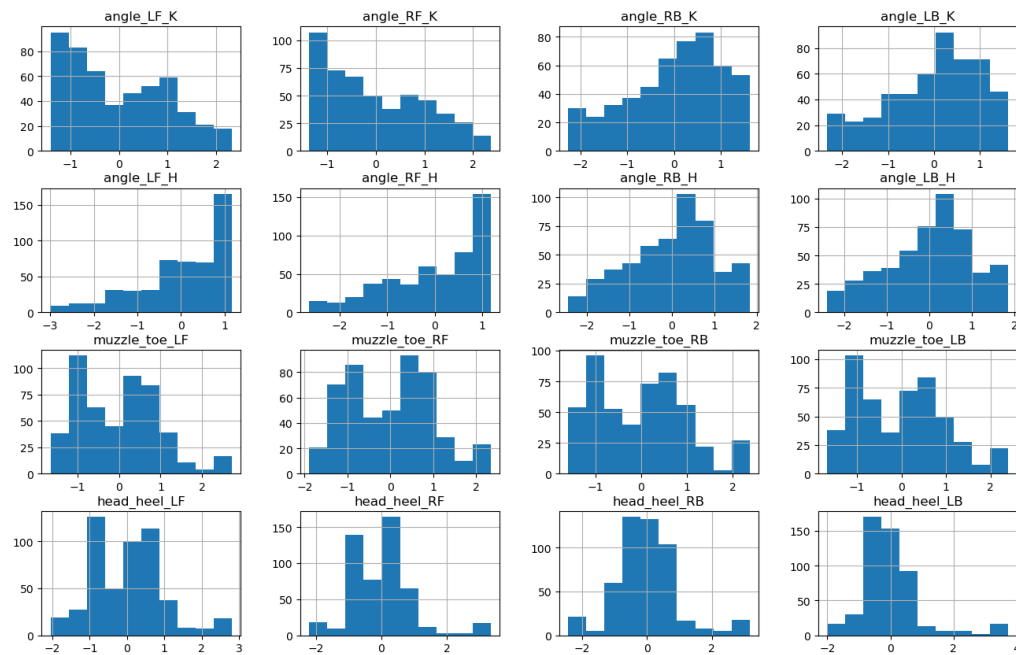


Figure 24 Distribution of all features after scaling

Appendix 8: Results

Regarding the models including all 11 actions, their accuracy (Figure 25), precision and recall (Figure 26), confusion matrixes (Figure 27) and a table of human-effort (Table 12) are shown. Finally, the dendrogram for the parent behaviours is shown (Figure 28).

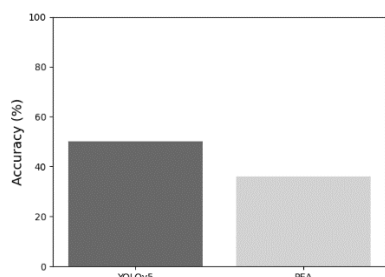


Figure 25 Overall accuracy YOLOv5 and PEA (PEA)

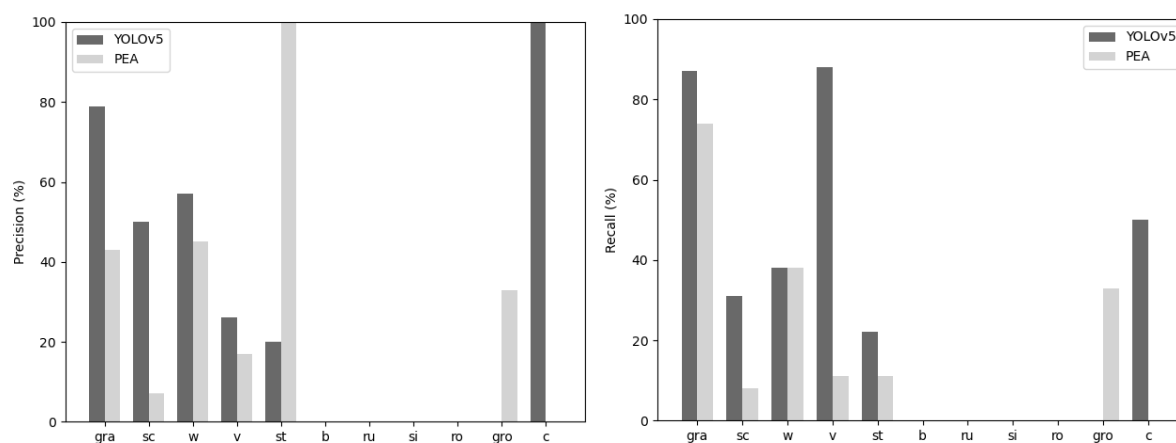


Figure 26 Precision (left) and recall (right) YOLOv5 and PEA (PEA)

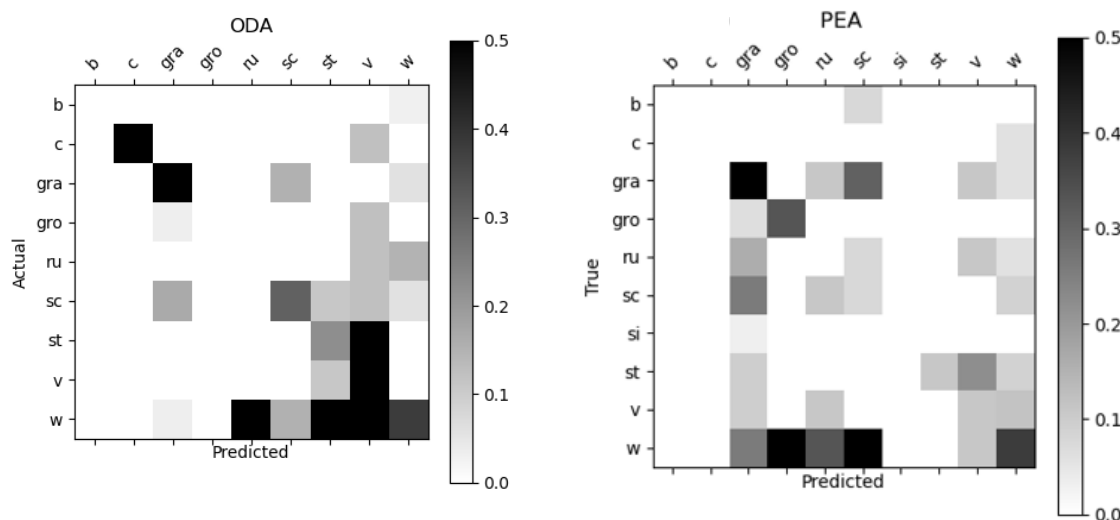


Figure 27 confusion matrix YOLOv5 (left) and PEA (PEA) (right). Predicted action (x-axis) and true labelled action (y-axis)

Table 12 Time effort YOLOv5 and PEA (PEA)

YOLOv5	Estimated time in hours
Data collection	22
Any Desk connection	1
Change csv to input BOX21	1
Set configuration files	1
Manually check bounding boxes	1
Train the network	1
Create confidence matrix	1
Inference	1
<i>Total</i>	29
PEA	Estimated building time in hours
Data collection	22
Any Desk connection	1
Preparation	6
label images	17
Creating training dataset	5
Prepare network	5
Analyse images	3
feature extraction	3
RF	1
Inference	1
<i>Total</i>	64

Finally, the PEA RF dendrogram is shown evaluating the parent actions (Figure 28). The dendrogram was created by running a single decision tree based on the output of the RF and used the Gini value as a measurement of the variation within a class, like MDI in the FIG. From the top to the left side of the dendrogram we see that two-distance head to leg features (muzzle_toe_RB & muzzle_toe_Lfw). If answer to the feature criteria was yes, the group went left and if not then to the right. After two splits there was already a homogenized group with 23 *foraging* images. From this dendrogram is becomes also clear that the leg angle features have not been able to create large, homogenized groups. Feature head_heel_Lfw was used but did not did only separate 7 samples. Due to an error in the plotting system the number of samples were incorrect as normally the sum of the values should equal the number of samples.

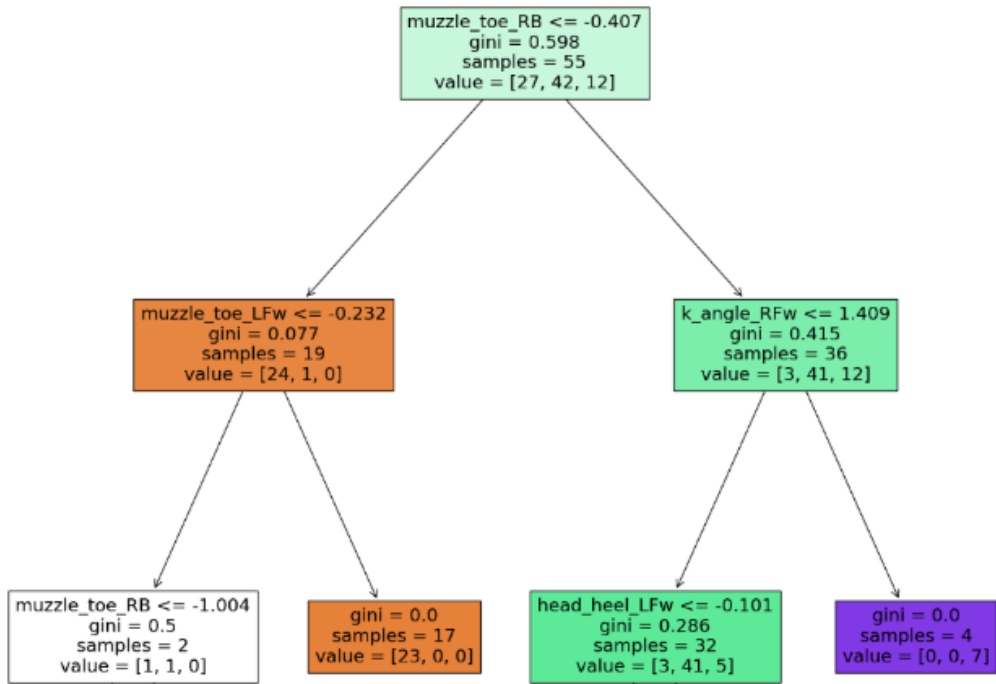


Figure 28 Decision tree based on predictions RF. Value = [Foraging, moving, other].